IBM

Personal Computer
Hardware Reference
Library

Technical Reference

Federal Communications Commission Radio Frequency Interference Statement

WARNING: This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception. If peripherals not offered by IBM are used with this equipment, it is suggested to use shielded grounded cables with in-line filters if necessary.

Notice: As sold by the manufacturer, the Prototype card does not require certification under the FCC's rules for Class B devices. The user is responsible for any interference to radio or TV reception which may be caused by a user-modified prototype card.

CAUTION: This product is equipped with a UL-listed and CSA-certified plug for the user's safety. It is to be used in conjunction with a properly grounded receptacle to avoid electrical shock.

Revised Edition (April 1983)

Changes are periodically made to the information herein; these changes will be incorporated in new editions of this publication.

Products are not stocked at the address below. Requests for copies of this product and for technical information about the system should be made to your authorized IBM Personal Computer dealer.

A Reader's Comment Form is provided at the back of this publication. If this form has been removed, address comments to: IBM Corp., Personal Computer, P.O. Box 1328-C, Boca Raton, Florida 33432. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligations whatever.

PREFACE

The IBM Personal Computer Technical Reference manual describes the hardware design and provides interface information for the IBM Personal Computer. This publication also has information about the basic input/output system (BIOS) and programming support.

The information in this publication is both introductory and for reference, and is intended for hardware and software designers, programmers, engineers, and interested persons who need to understand the design and operation of the computer.

You should be familiar with the use of the Personal Computer, and you should understand the concepts of computer architecture and programming.

This manual has two sections:

"Section 1: Hardware" describes each functional part of the system. This section also has specifications for power, timing, and interface. Programming considerations are supported by coding tables, command codes, and registers.

"Section 2: ROM BIOS and System Usage" describes the basic input/output system and its use. This section also contains the software interrupt listing, a BIOS memory map, descriptions of vectors with special meanings, and a set of low memory maps. In addition, keyboard encoding and usage is discussed.

The publication has seven appendixes:

Appendix A: ROM BIOS Listings
Appendix B: 8088 Assembly Instruction Set Reference
Appendix C: Of Characters, Keystrokes, and Color
Appendix D: Logic Diagrams
Appendix E: Specifications
Appendix F: Communications
Appendix G: Switch Settings

A glossary and bibliography are included.

Prerequisite Publication:

Guide to Operations for the IBM Personal Computer
Part Number 6025000

Suggested Reading:

BASIC for the IBM Personal Computer
Part Number 6025010

Disk Operating System (DOS) for the IBM Personal Computer
Part Number 6024061

Hardware Maintenance and Service for the IBM Personal Computer
Part Number 6025072

MACRO Assembler for the IBM Personal Computer
Part Number 6024002

Related publications are listed in the bibliography.

TABLE OF CONTENTS

Section 1: Hardware

IBM Personal Computer System Unit 1-3
IBM Personal Computer Math Coprocesser 1-33
IBM Keyboard 1-73
IBM Expansion Unit 1-79
IBM 80 CPS Printers 1-91
IBM Printer Adapter 1-117
IBM Monochrome Display and Printer Adapter 1-123
IBM Monochrome Display 1-131
IBM Color/Graphics Display Adapter 1-133
IBM Color Display 1-157
IBM 5-¼" Diskette Drive Adapter 1-159
IBM 5-¼" Diskette Drive 1-183
Diskettes 1-185
IBM Fixed Disk Drive Adapter 1-187
IBM 10MB Fixed Disk Drive 1-203
IBM Memory Expansion Options 1-205
IBM Game Control Adapter 1-211
IBM Prototype Card 1-217
IBM Asynchronous Communications Adapter 1-223
IBM Binary Synchronous Communications Adapter 1-251
IBM Synchronous Data Link Control (SDLC)
Communication Adapter 1-271
IBM Communications Adapter Cable 1-301

Section 2: ROM BIOS and System Usage

ROM BIOS 2-2
Keyboard Encoding and Usage 2-11
BIOS Cassette Logic 2-21

Appendix A: ROM BIOS Listings A-1

System BIOS A-2
Fixed Disk BIOS A-85

Appendix B: 8088 Assembly Instruction Set Reference B-1

Appendix C: Of Characters, Keystrokes, and Colors C-1

Appendix D: Logic Diagrams D-1

System Board (16/64K) D-2
System Board (64/256K) D-12
Keyboard - Type 1 .... D-22
Keyboard - Type 2 .... D-24
Expansion Board D-25
Extender Card D-26
Receiver Card D-29
Printer D-32
Printer Adapter D-35
Monochrome Display Adapter D-36
Color/Graphics Monitor Adapter D-46
Color Display D-52
Monochrome Display D-54
5-¼ Inch Diskette Drive Adapter D-55
5-¼ Inch Diskette Drive – Type 1 .... D-59
5-¼ Inch Diskette Drive – Type 2 .... D-62
Fixed Disk Drive Adapter D-64
Fixed Disk Drive – Type 1 .... D-70
Fixed Disk Drive – Type 2 .... D-73
32K Memory Expansion Option D-76
64K Memory Expansion Option D-79
64/256K Memory Expansion Option D-82
Game Control Adapter D-86
Prototype Card D-87
Asynchronous Communications Adapter D-88
Binary Synchronous Communications Adapter D-89
SDLC Communications Adapter D-91

Appendix E: Specifications E-1

Appendix F: Communications F-1

Appendix G: Switch Settings G-1

Glossary H-1

Index I-1

INDEX TAB LISTING

Section 1: Hardware ....

Section 2: ROM BIOS and System Usage ....

Appendix A: ROM BIOS Listings ....

Appendix B: 8088 Assembly Instruction Set Reference ....

Appendix C: Of Characters, Keystrokes, and Color ....

Appendix D: Logic Diagrams ....

Hardware

BIOS

Appendix A

Appendix B

Appendix C

Appendix D

Appendix E: Specifications ....

Appendix F: Communications ....

Appendix G: Switch Settings ....

Glossary ....

Bibliography ....

Index ....

Appendix E

Appendix F

Appendix G

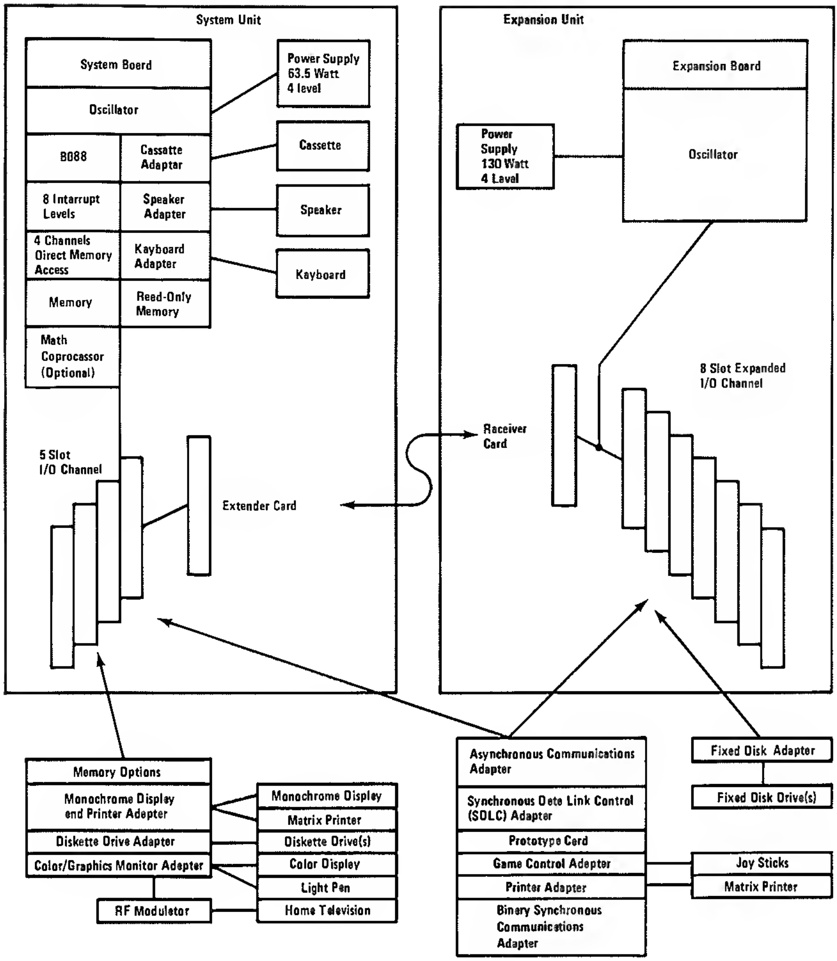
Glossary

Bibliography

Index

SECTION 1: HARDWARE

IBM Personal Computer System Unit 1-3
IBM Personal Computer Math Coprocesser 1-33
IBM Keyboard 1-73
IBM Expansion Unit 1-79
IBM 80 CPS Printers 1-91
IBM Printer Adapter 1-117
IBM Monochrome Display and Printer Adapter 1-123
IBM Monochrome Display 1-131
IBM Color/Graphics Display Adapter 1-133
IBM Color Display 1-157
IBM 5-¼" Diskette Drive Adapter 1-159
IBM 5-¼" Diskette Drive 1-183
Diskettes 1-185
IBM Fixed Disk Drive Adapter 1-187
IBM 10MB Fixed Disk Drive 1-203
IBM Memory Expansion Options 1-205
IBM Game Control Adapter 1-211
IBM Prototype Card 1-217
IBM Asynchronous Communications Adapter 1-223
IBM Binary Synchronous Communications Adapter 1-251
IBM Synchronous Data Link Control (SDLC)
Communication Adapter 1-271
IBM Communications Adapter Cable 1-301



System Block Diagram

IBM Personal Computer System Unit

The system unit is the standalone tabletop unit that contains the power supply, the speaker, and the system board.

The system unit contains one of two system boards. One system board supports 16K to 64K of read/write memory. The other system board supports 64K to 256K of read/write memory. Both system boards are functionally identical.

The power supply provides dc voltage to the system board and the internal drive(s).

System Board

The system board fits horizontally in the base of the system unit and is approximately 8-1/2 by 12 inches. It is a multilayer, single-land-per-channel design with ground and internal planes provided. DC power and a signal from the power supply enter the board through two six-pin connectors. Other connectors on the board are for attaching the keyboard, audio cassette, and speaker. Five 62-pin card edge-sockets are also mounted on the board. The I/O channel is bussed across these five I/O slots.

Two dual-in-line package (DIP) switches (two eight-switch packs) are mounted on the board and can be read under program control. The DIP switches provide the system software with information about the installed options, how much storage the system board has, what type of display adapter is installed, what operation modes are desired when power is switched on (color or black-and-white, 80- or 40-character lines), and the number of diskette drives attached.

The system board consists of five functional areas: the processor subsystem and its support elements, the read-only memory (ROM) subsystem, the read/write (R/W) memory subsystem, integrated I/O adapters, and the I/O channel. All are described in this section.

The heart of the system board is the Intel 8088 microprocessor. This processor is an 8-bit external bus version of Intel's 16-bit 8086 processor, and is software-compatible with the 8086. Thus, the 8088 supports 16-bit operations, including multiply and divide, and supports 20 bits of addressing (1 megabyte of storage). It also operates in maximum mode, so a co-processor can be added as a feature. The processor operates at a 4.77 MHz. This frequency, which is derived from a 14.31818-MHz crystal, is divided by 3 for the processor clock, and by 4 to obtain the 3.58-MHz color burst signal required for color televisions.

At the 4.77-MHz clock rate, the 8088 bus cycles are four clocks of 210 ns, or 840 ns. I/O cycles take five 210-ns clocks or 1.05 microseconds.

The processor is supported by a set of high-function support devices providing four channels of 20-bit direct-memory access (DMA), three 16-bit timer-counter channels, and eight prioritized interrupt levels.

Three of the four DMA channels are available on the I/O bus and support high-speed data transfers between I/O devices and memory without processor intervention. The fourth DMA channel is programmed to refresh the system dynamic memory. This is done by programming a channel of the timer-counter device to periodically request a dummy DMA transfer. This action creates a memory-read cycle, which is available to refresh dynamic storage both on the system board and in the system expansion slots. All DMA data transfers, except the refresh channel, take five processor clocks of 210 ns, or 1.05 μs if the processor-ready line is not deactivated. Refresh DMA cycles take four clocks or 840 ns.

The three programmable timer/counters are used by the system as follows: Channel 0 is used as a general-purpose timer providing a constant time base for implementing a time-of-day clock; Channel 1 is used to time and request refresh cycles from the DMA channel; and Channel 2 is used to support the tone generation for the audio speaker. Each channel has a minimum timing resolution of 1.05 us.

Of the eight prioritized levels of interrupt, six are bussed to the system expansion slots for use by feature cards. Two levels are used on the system board. Level 0, the highest priority, is attached to Channel 0 of the timer/counter and provides a periodic interrupt for the time-of-day clock. Level 1 is attached to the keyboard adapter circuits and receives an interrupt for each scan code sent by the keyboard. The non-maskable interrupt (NMI) of the 8088 is used to report memory parity errors.

The system board supports both ROM and R/W memory. It has space for 48K x 8 of ROM or EPROM. Six module sockets are provided, each of which can accept an 8K by 8 byte device. Five of the sockets are populated with 40K bytes of ROM. This ROM contains the cassette BASIC interpreter, cassette operating system, power-on self-test, I/O drivers, dot patterns for 128 characters in graphics mode, and a diskette bootstrap loader. The ROM is packaged in 24-pin modules and has an access time of 250 ns and a cycle time of 375 ns.

The difference between the R/W memory on the two system boards is shown in the following chart.

System Board	Minimum Storage	Maximum Storage	Memory Modules	Soldered (Bank 0)	Pluggable (Bank 1-3)
16/64K	16K	64K	16K by 1 Bit	1 Bank of 9	3 Banks of 9
64/256K	64K	256K	64K by 1 Bit	1 Bank of 9	3 Banks of 9

Memory greater than either system board's maximum is obtained by adding memory cards in the expansion slots. All memory is parity-checked and consists of dynamic 16K by 1 bit or (64K by 1 bit) chips with an access time of 250 ns and a cycle time of 410 ns.

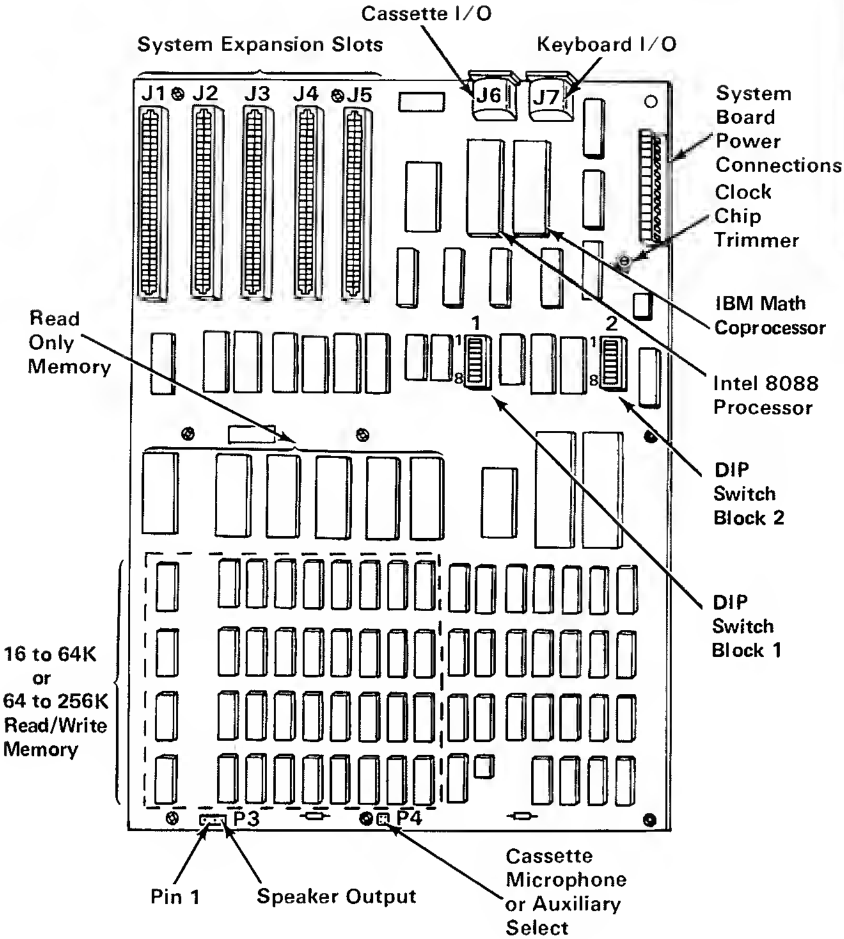
The system board contains circuits for attaching an audio cassette, the keyboard, and the speaker. The cassette adapter allows the attachment of any good quality audio cassette through the earphone output and either the microphone or auxiliary inputs. The system board has a jumper for either input. This interface also provides a cassette motor control line for transport starting and stopping under program control. This interface reads and writes the audio cassette at a data rate of between 1,000 and 2,000 baud. The baud rate is variable and dependent on data content, because a different bit-cell time is used for 0's and 1's. For diagnostic purposes, the tape interface can loop read to write for testing the system board's circuits. The ROM cassette software blocks cassette data and generates a cyclic redundancy check (CRC) to check this data.

The system board contains the adapter circuits for attaching the serial interface from the keyboard. These circuits generate an interrupt to the processor when a complete scan code is received. The interface can request execution of a diagnostic test in the keyboard.

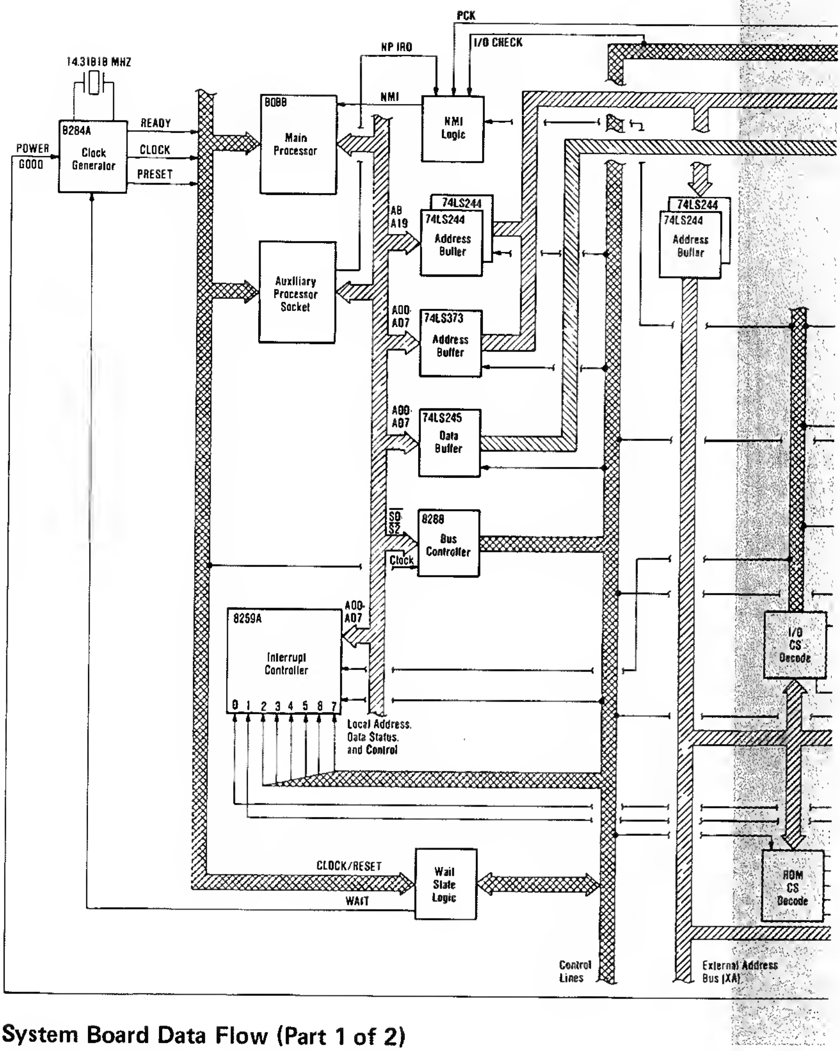
Both the keyboard and cassette interfaces are 5-pin DIN connectors on the system board that extend through the rear panel of the system unit.

The system unit has a 2-1/4 inch audio speaker. The speaker's control circuits and driver are on the system board. The speaker connects through a 2-wire interface that attaches to a 3-pin connector on the system board.

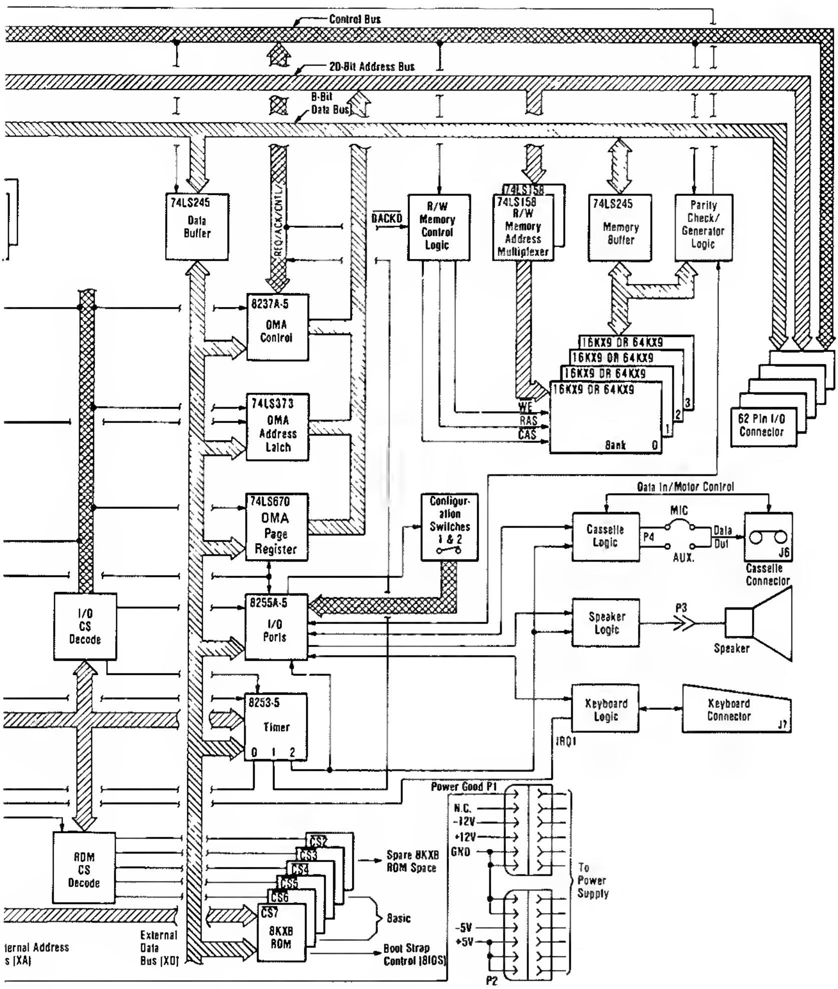
The speaker drive circuit is capable of approximately 1/2 watt of power. The control circuits allow the speaker to be driven three different ways: 1.) a direct program control register bit may be toggled to generate a pulse train; 2.) the output from Channel 2 of the timer counter may be programmed to generate a waveform to the speaker; 3.) the clock input to the timer counter can be modulated with a program-controlled I/O register bit. All three methods may be performed simultaneously.



System Board Component Diagram



System Board Data Flow (Part 1 of 2)



System Board Data Flow (Part 2 of 2)

Hex Range	Usage
000-00F	DMA Chip 8237A-5
020-021	Interrupt 8259A
040-043	Timer 8253-5
060-063	PPI 8255A-5
080-083	DMA Page Registers
0Ax*	NMI Mask Register
0Cx	Reserved
0Ex	Reserved
100-1FF	Not Usable
200-20F	Game Control
210-217	Expansion Unit
220-24F	Reserved
278-27F	Reserved
2F0-2F7	Reserved
2F8-2FF	Asynchronous Communications (Secondary)
300-31F	Prototype Card
320-32F	Fixed Disk
378-37F	Printer
380-38C**	SDLC Communications
380-389**	Binary Synchronous Communications (Secondary)
3A0-3A9	Binary Synchronous Communications (Primary)
380-38F	I8M Monochrome Display/Printer
3C0-3CF	Reserved
3D0-3DF	Color/Graphics
3E0-3F7	Reserved
3F0-3F7	Diskette
3F8-3FF	Asynchronous Communications (Primary)

* At power-on time, the Non Mask Interrupt into the 8088 is masked off. This mask bit can be set and reset through system software as follows:

Set mask: Write hex 80 to I/O Address hex A0 (enable NMI)

Clear mask: Write hex 00 to I/O Address hex A0 (disable NMI)

** SDLC Communications and Secondary Binary Synchronous Communications cannot be used together because their hex addresses overlap.

I/O Address Map

Number	Usage
NMI	Parity
0	Timer
1	Keyboard
2	Reserved
3	Asynchronous Communications (Secondary)
	SDLC Communications
	BSC (Secondary)
4	Asynchronous Communications (Primary)
	SDLC Communications
	BSC (Primary)
5	Fixed Disk
6	Diskette
7	Printer

8088 Hardware Interrupt Listing

Hex Port Number		Bit				
0060	INPUT	PA0	+Keyboard Scan Code 0	Or	IPL 5-1/4 Diskette Drive	(SW1—1)
		1	1		Reserved	(SW1—2)
		2	2		System Board Read/Write Memory Size	*(SW1—3)
		3	3		System Board Read/Write Memory Size	*(SW1—4)
		4	4		+Display Type 1	**(SW1—5)
		5	5		+Display Type 2	**(SW1—6)
		6	6		No. of 5-1/4 Drives	***(SW1—7)
		7	7		No. of 5-1/4 Drives	***(SW1—8)
0061	OUTPUT	PB0	+Timer 2 Gate Speaker			
		1	+Speaker Data			
		2	+(Read Read/Write Memory Size) or (Read Spare Key)			
		3	+Cassette Motor Off			
		4	-Enable Read/Write Memory			
		5	-Enable I/O Channel Check			
		6	-Hold Keyboard Clock Low			
		7	-(Enable Keyboard) or + (Clear Keyboard and Enable Sense Switches)			
0062	INPUT	PC0	I/O Read/Write Memory (Sw2—1)	Binary Value X 32K	Or	I/O Read/Write Memory (Sw2—5)
		1	I/O Read/Write Memory (Sw2—2)			
		2	I/O Read/Write Memory (Sw2—3)			
		3	I/O Read/Write Memory (Sw2—4)			
		4	+Cassette Data In			
		5	+Timer Channel 2 Out			
		6	+I/O Channel Check			
		7	+Read/Write Memory Parity Check			

0063 Command/Mode Register

Mode Register Value — Hex 99

7	6	5	4	3	2	1	0
1	0	0	1	1	0	0	1

* PA3 Sw1—4	PA2 Sw1—3	Amount of Memory Located on System Board
0	0	16K
0	1	32K
1	0	48K
1	1	64 to 256K

** PA5 Sw1—6	PA4 Sw1—5	Display at Power-Up Mode
0	0	Reserved
0	1	Color 40 X 25 (BW Mode)
1	0	Color 80 X 25 (BW Mode)
1	1	IBM Monochrome (80 X 25)

*** PA7 Sw1—8	PA6 Sw1—7	Number of 5-1/4" Drives in System
0	0	1
0	1	2
1	0	3
1	1	4

Note: A plus (+) indicates a bit value of 1 performs the specified function.
A minus (-) indicates a bit value of 0 performs the specified function.
PA Bit = 0 implies switch "ON." PA bit = 1 implies switch "OFF."

8255A I/O Bit Map

Start Address Decimal	Hex	Function
0	00000	16 to 64K Read/Write Memory on System Board
16K	04000	
32K	08000	
48K	0C000	
64K	10000	Up to 576K Read/Write Memory in I/O Channel
80K	14000	
96K	18000	
112K	1C000	
128K	20000	
144K	24000	
160K	28000	
176K	2C000	
192K	30000	
208K	34000	
224K	38000	
240K	3C000	
256K	40000	
272K	44000	
288K	48000	
304K	4C000	
320K	50000	
336K	54000	
352K	58000	
368K	5C000	
384K	60000	
400K	64000	
416K	68000	
432K	6C000	
448K	70000	
464K	74000	
480K	78000	
496K	7C000	
512K	80000	
528K	84000	
544K	88000	
560K	8C000	
576K	90000	
592K	94000	
608K	98000	
624K	9C000	

System Memory Map for 16/64K System Board (Part 1 of 2)

Start Address Decimal	Start Address Hex	Function
640K	A0000	128K Reserved
656K	A4000	
672K	A8000	
688K	AC000	
704K	B0000	Monochrome
720K	B4000	
736K	B8000	Color/Graphics
752K	BC000	
768K	C0000	
784K	C4000	
800K	C8000	Fixed Disk Control
816K	CC000	192K Read Only Memory Expansion and Control
832K	D0000	
848K	D4000	
864K	D8000	
880K	DC000	
896K	E0000	
912K	E4000	
928K	E8000	
944K	EC000	
960K	F0000	Reserved
976K	F4000	48K Base System ROM
992K	F8000	
1008K	FC000	

System Memory Map for 16/64K System Board (Part 2 of 2)

Start Address Decimal	Hex	Function
0	00000	64 to 256K Read/Write Memory on System Board
16K	04000	
32K	08000	
48K	0C000	
64K	10000	
80K	14000	
96K	18000	
112K	1C000	
128K	20000	
144K	24000	
160K	28000	
176K	2C000	
192K	30000	
208K	34000	
224K	38000	
240K	3C000	
256K	40000	Up to 384K Read/Write Memory in I/O Channel Up to 384K in I/O Channel
272K	44000	
288K	48000	
304K	4C000	
320K	50000	
336K	54000	
352K	58000	
368K	5C000	
384K	60000	
400K	64000	
416K	68000	
432K	6C000	
448K	70000	
464K	74000	
480K	78000	
496K	7C000	
512K	80000	
528K	84000	
544K	88000	
560K	8C000	
576K	90000	
592K	94000	
608K	98000	
624K	9C000	

System Memory Map for 64/256K System Board (Part 1 of 2)

Start Address Decimal	Hex	Function
640K	A0000	128K Reserved
656K	A4000	
672K	A8000	
688K	AC000	
704K	B0000	Monochrome
720K	B4000	
736K	B8000	Color/Graphics
752K	BC000	
768K	C0000	192K Read Only Memory Expansion and Control
784K	C4000	
800K	C8000	Fixed Disk Control
816K	CC000	
832K	D0000	
848K	D4000	
864K	D8000	
880K	DC000	
896K	E0000	
912K	E4000	
928K	E8000	
944K	EC000	
960K	F0000	Reserved
976K	F4000	48K Base System ROM
992K	F8000	
1008K	FC000	

System Memory Map for 64/256K System Board (Part 2 of 2)

System Board Switch Settings

All system board switch settings for total system memory, number of diskette drives, and type of display adapter are located in "Appendix G: Switch Settings."

I/O Channel

The I/O channel is an extension of the 8088 microprocessor bus. It is, however, demultiplexed, repowered, and enhanced by the addition of interrupts and direct memory access (DMA) functions.

The I/O channel contains an 8-bit, bidirectional data bus, 20 address lines, 6 levels of interrupt, control lines for memory and I/O read or write, clock and timing lines, 3 channels of DMA control lines, memory refresh timing control lines, a channel-check line, and power and ground for the adapters. Four voltage levels are provided for I/O cards: +5 Vdc, −5 Vdc, +12 Vdc, and −12 Vdc. These functions are provided in a 62-pin connector with 100-mil card tab spacing.

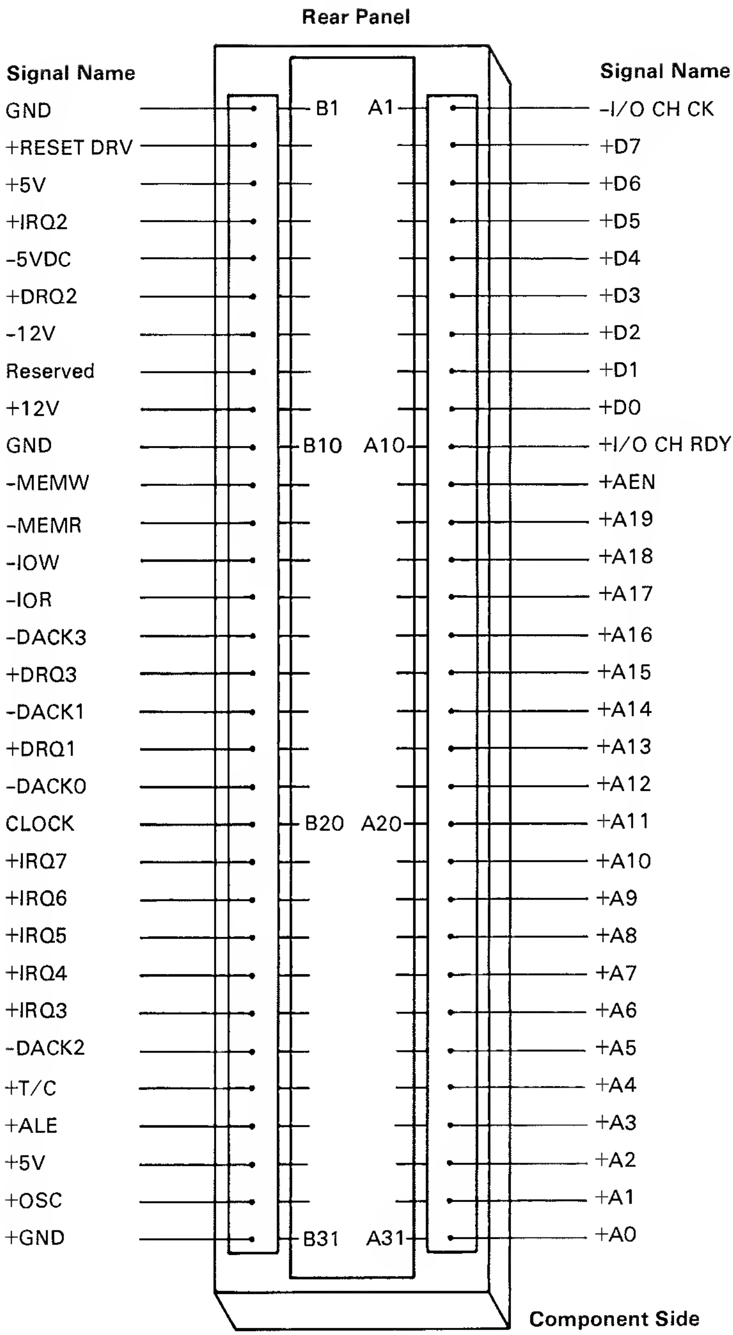
A 'ready' line is available on the I/O channel to allow operation with slow I/O or memory devices. If the channel's ready line is not activated by an addressed device, all processor-generated memory read and write cycles take four 210-ns clock or 840-ns/byte. All processor-generated I/O read and write cycles require five clocks for a cycle time of 1.05 μs/byte. All DMA transfers require five clocks for a cycle time of 1.05 μs/byte. Refresh cycles occur once every 72 clocks (approximately 15 μs) and require four clocks or approximately 7% of the bus bandwidth.

I/O devices are addressed using I/O mapped address space. The channel is designed so that 512 I/O device addresses are available to the I/O channel cards.

A 'channel check' line exists for reporting error conditions to the processor. Activating this line results in a Non-Maskable Interrupt (NMI) to the 8088 processor. Memory expansion options use this line to report parity errors.

The I/O channel is repowered to provide sufficient drive to power all five system unit expansion slots, assuming two low-power Schottky loads per slot. The IBM I/O adapters typically use only one load.

The following pages describe the system board's I/O channel.



I/O Channel Diagram

I/O Channel Description

The following is a description of the IBM Personal Computer I/O Channel. All lines are TTL-compatible.

Signal	I/O	Description
OSC	O	Oscillator: High-speed clock with a 70-ns period (14.31818 MHz). It has a 50% duty cycle.
CLK	O	System clock: It is a divide-by-three of the oscillator and has a period of 210 ns (4.77 MHz). The clock has a 33% duty cycle.
RESET DRV	O	This line is used to reset or initialize system logic upon power-up or during a low line voltage outage. This signal is synchronized to the falling edge of clock and is active high.
A0-A19	O	Address bits 0 to 19: These lines are used to address memory and I/O devices within the system. The 20 address lines allow access of up to 1 megabyte of memory. A0 is the least significant bit (LSB) and A19 is the most significant bit (MSB). These lines are generated by either the processor or DMA controller. They are active high.
D0-D7	I/O	Data Bits 0 to 7: These lines provide data bus bits 0 to 7 for the processor, memory, and I/O devices. D0 is the least significant bit (LSB) and D7 is the most significant bit (MSB). These lines are active high.

Signal	I/O	Description
ALE	O	Address Latch Enable: This line is provided by the 8288 Bus Controller and is used on the system board to latch valid addresses from the processor. It is available to the I/O channel as an indicator of a valid processor address (when used with AEN). Processor addresses are latched with the failing edge of ALE.
$\overline{\text{I/O CH CK}}$	I	-I/O Channel Check: This line provides the processor with parity (error) information on memory or devices in the I/O channel. When this signal is active low, a parity error is indicated.
I/O CH RDY	I	I/O Channel Ready: This line, normally high (ready), is pulled low (not ready) by a memory or I/O device to lengthen I/O or memory cycles. It allows slower devices to attach to the I/O channel with a minimum of difficulty. Any slow device using this line should drive it low immediately upon detecting a valid address and a read or write command. This line should never be held low longer than 10 clock cycles. Machine cycles (I/O or memory) are extended by an integral number of CLK cycles (210 ns).
IRQ2-IRQ7	I	Interrupt Request 2 to 7: These lines are used to signal the processor that an I/O device requires attention. They are prioritized with IRQ2 as the highest priority and IRQ7 as the lowest. An Interrupt Request is generated by raising an IRQ line (low to high) and holding it high until it is acknowledged by the processor (interrupt service routine).

Signal	I/O	Description
$\overline{\text{IOR}}$	O	-I/O Read Command: This command line instructs an I/O device to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{IOW}}$	O	-I/O Write Command: This command line instructs an I/O device to read the data on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{MEMR}}$	O	Memory Read Command: This command line instructs the memory to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{MEMW}}$	O	Memory Write Command: This command line instructs the memory to store the data present on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
DRQ1-DRQ3	I	DMA Request 1 to 3: These lines are asynchronous channel requests used by peripheral devices to gain DMA service. They are prioritized with DRQ3 being the lowest and DRQ1 being the highest. A request is generated by bringing a DRQ line to an active level (high). A DRQ line must be held high until the corresponding DACK line goes active.
$\overline{\text{DACK0-}}$ $\overline{\text{DACK3}}$	O	-DMA Acknowledge 0 to 3: These lines are used to acknowledge DMA requests (DRQ1-DRQ3) and to refresh system dynamic memory (DACK0). They are active low.

Signal	I/O	Description
AEN	O	Address Enable: This line is used to de-gate the processor and other devices from the I/O channel to allow DMA transfers to take place. When this line is active (high), the DMA controller has control of the address bus, data bus, read command lines (memory and I/O), and the write command lines (memory and I/O).
T/C	O	Terminal Count: This line provides a pulse when the terminal count for any DMA channel is reached. This signal is active high.

The following voltages are available on the system board I/O channel:

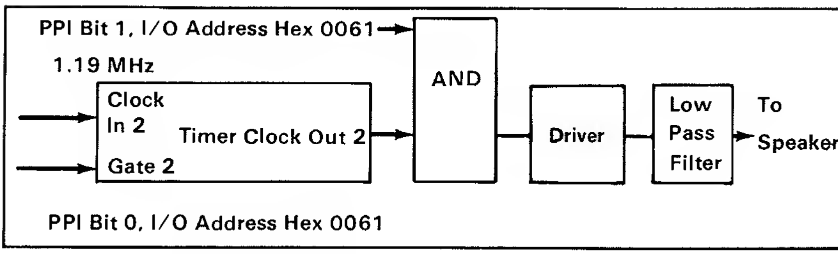
+5 Vdc ±5%, located on 2 connector pins
−5 Vdc ±10%, located on 1 connector pin
+12 Vdc ±5%, located on 1 connector pin
−12 Vdc ±10%, located on 1 connector pin
GND (Ground), located on 3 connector pins

Speaker Interface

The sound system has a small, permanent-magnet, 2-1/4 inch speaker. The speaker can be driven from one or both of two sources:

- An 8255A-5 PPI output bit. The address and bit are defined in the "I/O Address Map."

- A timer clock channel, the output of which is programmable within the functions of the 8253-5 timer when using a 1.19-MHz clock input. The timer gate also is controlled by an 8255A-5 PPI output-port bit. Address and bit assignment are in the "I/O Address Map."



Speaker Drive System Block Diagram

Channel 2 (Tone generation for speaker)		
	Gate 2	— Controller by 8255A-5 PPI Bit (See I/O Map)
	Clock In 2	— 1.19318 - MHz OSC
	Clock Out 2	— Used to drive speaker

Speaker Tone Generation

The speaker connection is a 4-pin Berg connector. See "System Board Component Diagram," earlier in this section, for speaker connection or placement.

Pin	Function
1	Data
2	Key
3	Ground
4	+5 Volts

Speaker Connector

Power Supply

The system power supply is located at the right rear of the system unit. It is designed to be an integral part of the system-unit chassis. Its housing provides support for the rear panel, and its fan furnishes cooling for the whole system.

It supplies the power and reset signal necessary for the operation of the system board, installable options, and the keyboard. It also provides a switched ac socket for the IBM Monochrome Display and two separate connectors for power to the 5-1/4 inch diskette drives.

It is a dc-switching power supply designed for continuous operation at 63.5 watts. It has a fused 120-Vac input and provides four regulated dc output voltages: 7 A of +5 Vdc, 2 A of +12 Vdc, 0.3 A of −5 Vdc, and 0.25 A of −12 Vdc. These outputs are over-voltage, over-current, open-circuit, and short-circuit protected. If a dc overload or over-voltage condition occurs, all dc outputs are shut down as long as the condition exists.

The +5 Vdc powers the logic on the system board and the diskette drives and allows approximately 4 A of +5 Vdc for the adapters in the system-unit expansion slots. The +12 Vdc power level is designed to power the system's dynamic memory and the two internal 5-1/4 inch diskette drive motors. It is assumed that only one drive is active at a time. The −5 Vdc level is designed for dynamic memory bias voltage; it tracks the +5 Vdc and +12 Vdc very quickly at power-on and has a longer decay on power-off than the +5 Vdc and +12 Vdc outputs. The +12 Vdc and −12 Vdc are used for powering the EIA drivers on the communications adapters. All four power levels are bussed across the five system-unit expansion slots.

Operating Characteristics

Input Requirements

The following are the input requirements for the system unit power supply.

Voltage (Vac)			Frequency (Hz)	Current (Amps)
Nominal	Minimum	Maximum	+/- 3Hz	Maximum
120	104	127	60	2.5 at 104 Vac

Vdc Output

The following are the dc outputs for the system unit power supply.

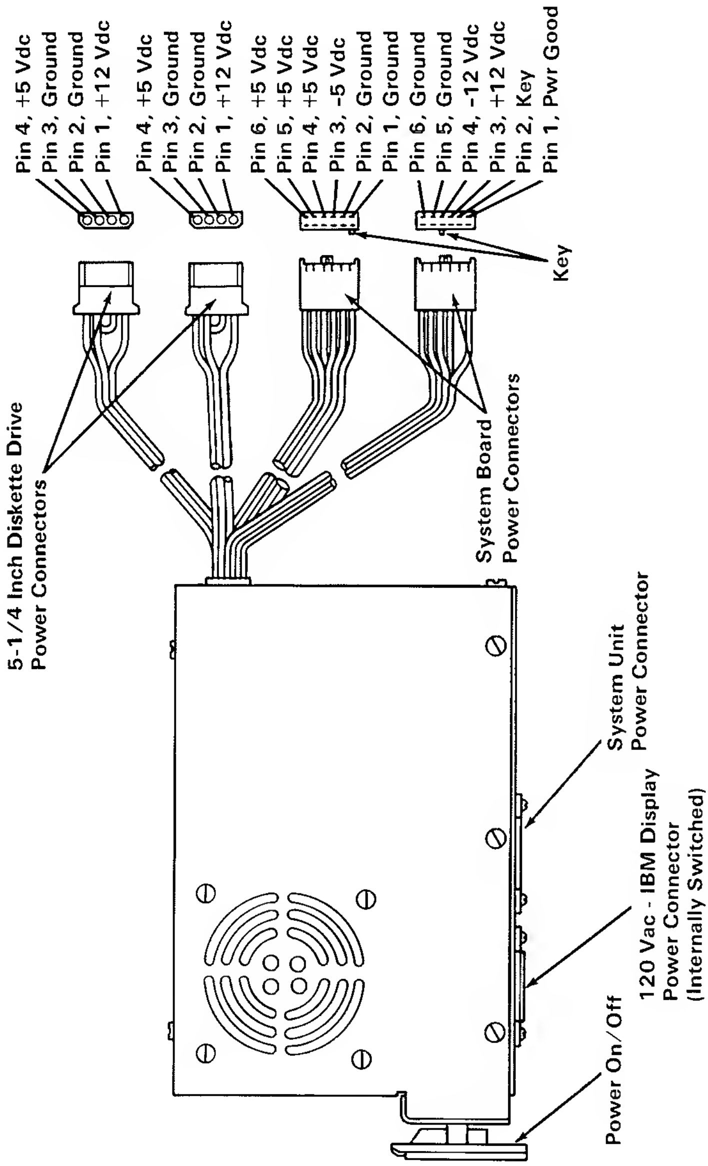
Voltage (Vdc)	Current (Amps)		Regulation (Tolerance)	
Nominal	Minimum	Maximum	+%	-%
+5.0	2.3	7.0	5	4
-5.0	0.0	0.3	10	8
+12.0	0.4	2.0	5	4
-12.0	0.0	0.25	10	9

Vac Output

The power supply provides a filtered, ac output that is switched on and off with the main power switch. The maximum current available at this output is 0.75 A. The recepticle provided at the rear of the power supply for this ac output is a nonstandard connector designed to be used only for the IBM Monochrome Display.

Power Supply Connectors and Pin Assignments

The power connector on the system board is a 12-pin male connector that plugs into the power-supply connectors. The pin configurations and locations are shown below:



Power Supply and Connectors

Over-Voltage/Over-Current Protection

The system power supply employs protection features which are described below.

Primary (Input)

The following table describes the primary (input voltage) protection for the system-unit power supply.

Voltage (Nominal Vac)	Type Protection	Rating (Amps)
120	Fuse	2

Secondary (Output)

On over-voltage, the power supply is designed to shut down all outputs when either the +5 Vdc or the +12 Vdc output exceeds 200% of its maximum rated voltage. On over-current, the supply will turn off if any output exceeds 130% of its nominal value.

Power-Good Signal

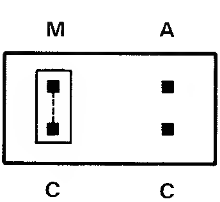
When the power supply is turned on after it has been off for a minimum of 5 seconds, it generates a power-good signal which indicates that there is adequate power for processing. When the four output voltages are above the minimum sense levels, as described below, the signal sequences to a TTL-compatible up level (2.4 Vdc to 5.5 Vdc), which is capable of sourcing 60 μA. When any of the four output voltages is below its minimum sense level or above its maximum sense level, the power good signal will be a TTL-compatible down level (0.0 Vdc to 0.4 Vdc) capable of sourcing 500 μA. The power good signal has a turn-on delay of 100 ms after the output voltages have reached their respective minimum sense levels.

Output Voltage	Under-Voltage Nominal Sense Level	Over-Voltage Nominal Sense Level
+5 Vdc	+4.0 Vdc	+5.9 Vdc
-5 Vdc	-4.0 Vdc	-5.9 Vdc
+12 Vdc	+9.6 Vdc	+14.2 Vdc
-12 Vdc	-9.6 Vdc	-14.2 Vdc

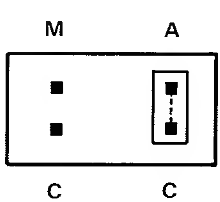
Cassette Interface

The cassette interface is controlled through software. An output from the 8253 timer controls the data to the cassette recorder through pin 5 of the cassette DIN connector at the rear of the system board. The cassette input data is read by an input port bit of the 8255A-5 programmable peripheral interface (8255A-5 PPI). This data is received through pin 4 of the cassette connector. Software algorithms are used to generate and read cassette data. The cassette drive motor is controlled through pins 1 and 3 of the cassette connector. The drive motor on/off switching is controlled by an 8255A-5 PPI output-port bit (hex 61, bit 3). The 8255A-5 address and bit assignments are defined in "I/O Address Map" earlier in this section.

A 2 by 2 Berg pin and a jumper are used on the cassette 'data out' line. The jumper allows use of the 'data out' line as a 0.075-Vdc microphone input when placed across the M and C pins of the Berg connector. A 0.68-Vdc auxiliary input to the cassette recorder is available when the jumper is placed across the A and C pins of the Berg connector. The "System Board Component Diagram" shows the location of the cassette Berg pins.



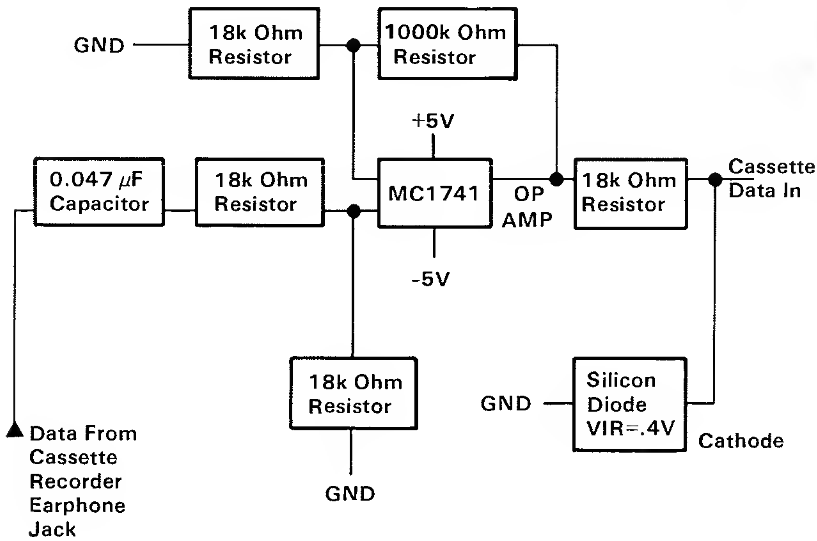
Microphone Input
(0.075 Vdc)



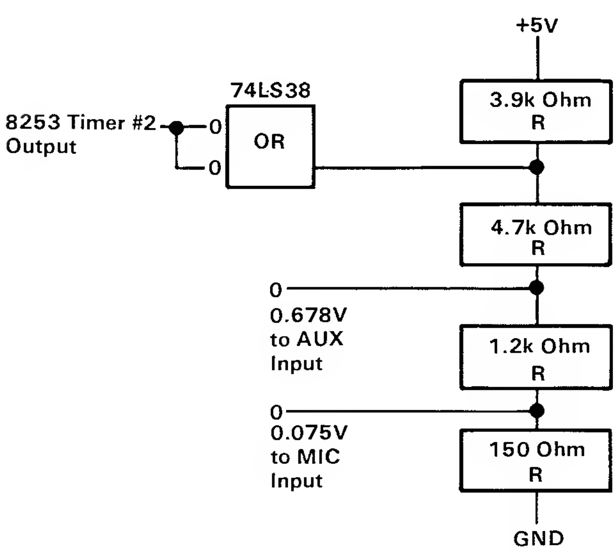
Auxiliary Input
(0.68 Vdc)

Cassette Circuit Block Diagrams

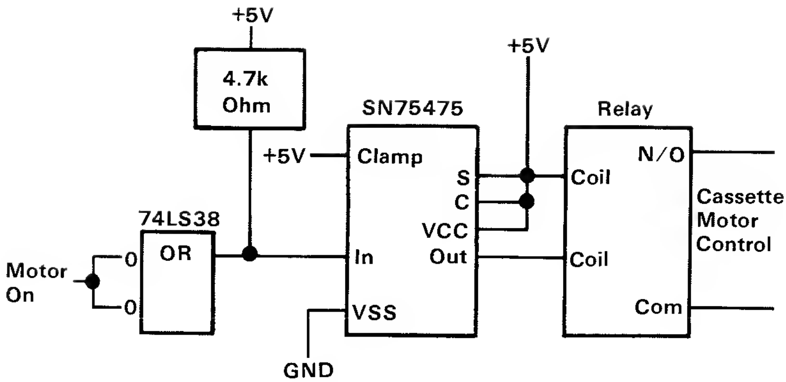
Circuit block diagrams for the cassette-interface read hardware, write hardware, and motor control are illustrated below.



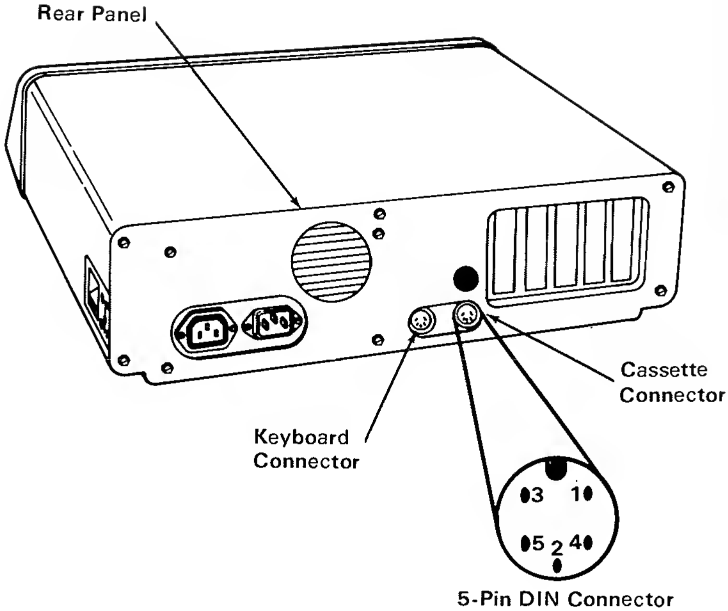
Cassette Interface Read Hardware Block Diagram



Cassette Interface Write Hardware Block Diagram



Cassette Motor Control Block Diagram



Pin	Signal	Electrical Characteristics
1	Motor Control	Common from Relay
2	Ground	
3	Motor Control	Relay N.O. (6 Vdc at 1A)
4	Data In	500nA at ±13V - at 1,000 - 2,000 Baud
5	Data Out (Microphone or Auxiliary)	250 μA at 0.68 Vdc or ** 0.075 Vdc

*All voltages and currents are maximum ratings and should not be exceeded.

**Data out can be chosen using a jumper located on the system board. (Auxiliary → 0.68 Vdc or Microphone → 0.075 Vdc).

Interchange of these voltages on the cassette recorder could lead to damage of recorder inputs.

Cassette Interface Connector Specifications

Notes:

IBM Personal Computer Math Coprocessor

The IBM Personal Computer Math Coprocessor enables the IBM Personal Computer to perform high speed arithmetic, logarithmic functions, and trigonometric operations with extreme accuracy.

The coprocessor works in parallel with the processor. The parallel operation decreases operation time by allowing the coprocessor to do mathematical calculations while the processor continues to do other functions.

The first five bits of every instruction opcode for the coprocessor are identical (11011 binary). When the processor and the coprocessor see this instruction opcode, the processor calculates the address, of any variables in memory, while the coprocessor checks the instruction. The coprocessor will then take the memory address from the processor if necessary. To access locations in memory, the coprocessor takes the local bus from the processor when the processor finishes its current instruction. When the coprocessor is finished with the memory transfer, it returns the local bus to the processor.

The IBM Math Coprocessor works with seven numeric data types divided into the three classes listed below.

- Binary integers (3 types)
- Decimal integers (1 type)
- Real numbers (3 types)

Programming Interface

The coprocessor extends the data types, registers, and instructions to the processor.

The coprocessor has eight 80-bit registers which provide the equivalent capacity of 40 16-bit registers found in the processor. This register space allows constants and temporary results to be held in registers during calculations, thus reducing memory access and improving speed as well as bus availability. The register space can be used as a stack or as a fixed register set. When used as a stack, only the top two stack elements are operated on: when used as a fixed register set, all registers are operated on. The Figure below shows representations of large and small numbers in each data type.

Data Type	Bits	Significant Digits (Decimal)	Approximate Range (decimal)
Word Integer	16	4	$-32,768 \leq X \leq +32,767$
Short Integer	32	9	$-2x10^9 \leq X \leq +2x10^9$
Long Integer	64	18	$-9x10^{18} \leq X \leq +9x10^{18}$
Packed Decimal	80	18	$-99...99 \leq X \leq +99...99$ (18 digits)
Short Real*	32	6-7	$8.43x10^{-37} \leq \lvert X \rvert \leq 3.37x10^{38}$
Long Real*	64	15-16	$4.19x10^{-307} \leq \lvert X \rvert \leq 1.67x10^{308}$
Temporary Real	80	19	$3.4x10^{-4932} \leq \lvert X \rvert \leq 1.2x10^{4932}$

*The short and long real data types correspond to the single and double precision data types

Data Types

Hardware Interface

The coprocessor utilizes the same clock generator and system bus interface components as the processor. The coprocessor is wired directly into the processor, as shown in the coprocessor interconnection diagram. The processor's queue status lines (QS0 and QS1) enable the coprocessor to obtain and decode instructions simultaneously with the processor. The coprocessor's busy signal informs the processor that it is executing; the processor's WAIT instruction forces the processor to wait until the coprocessor is finished executing (wait for NOT BUSY).

When an incorrect instruction is sent to the coprocessor (for example; divide by zero or load a full register), the coprocessor can signal the processor with an interrupt. There are three conditions that will disable the coprocessor interrupt to the processor:

1. Exception and Interrupt Enable bits of the control word are set to 1's.

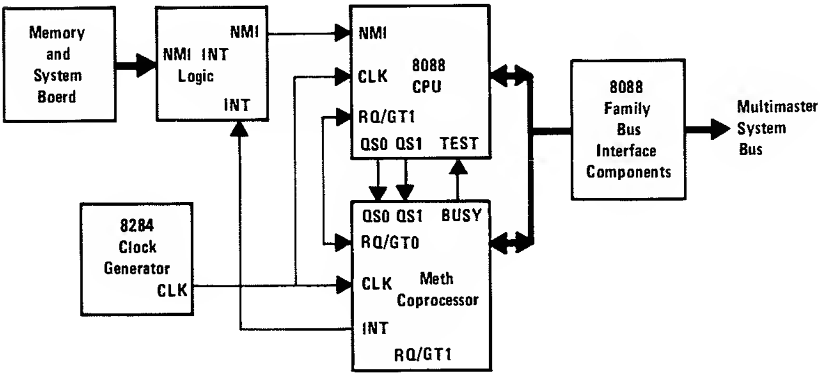
2. System board switch block 1 switch 2 set in the On position.

3. NMI Mask REG is set to zero.

At power-on time the NMI Mask REG is cleared to disable the NMI. Any software using the coprocessor's interrupt capability must ensure that conditions 2 and 3 are never met during the operation of the software or an "Endless Wait" will occur. An "Endless Wait" will have the processor waiting for the "Not Busy" signal from the coprocessor while the coprocessor is waiting for the processor to interrupt.

Because a memory parity error may also cause an interrupt to the 8088 NMI line, the program should check that a parity error did not occur (by reading the 8255 port), then clear exceptions by executing the FNSAVE or the FNCLEX instruction. In most cases, the status word would be looked at, and the exception would be identified and acted upon.

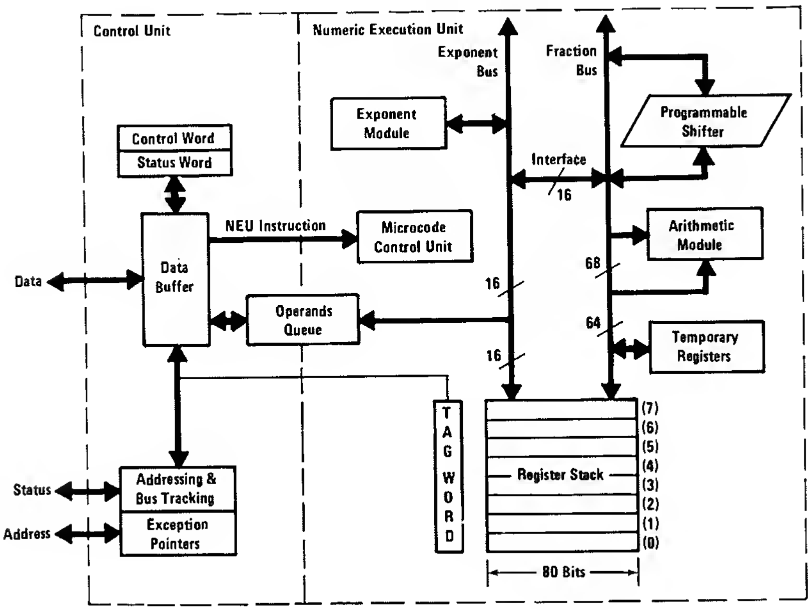
The NMI Mask REG and the coprocessors interrupt are tied to the NMI line through the NMI interrupt logic. Minor conversions of software designed for use with an 8087 must be made before existing software will be compatible with the IBM Personal Computer Math Coprocessor.



Coprocessor Interconnection

Control Unit

The control unit (CU) of the coprocessor and the processor fetch all instructions at the same time, as well as every byte of the instruction stream at the same time. The simultaneous fetching allows the coprocessor to know what the processor is doing at all times. This is necessary to keep a coprocessor instruction from going unnoticed. Coprocessor instructions are mixed with processor instructions in a single data stream. To aid the coprocessor in tracking the processor, nine status lines are interconnected (QS0, QS1, and S0 through S6).



Coprocessor Block Diagram

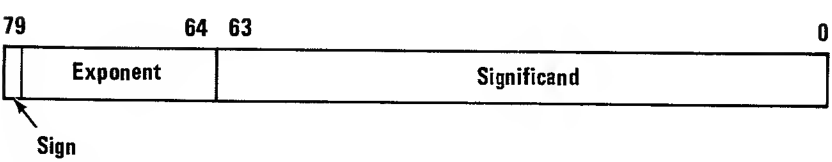
Register Stack

Each of the eight registers in the coprocessor's register stack is 80 bits wide, and each is divided into the "fields" shown in the figure below. The format in the figure below corresponds to the coprocessor's temporary real data type that is used for all calculations.

The ST field in the status word identifies the current top-of-stack register. A load ("push") operation decreases ST by 1 and loads a new value into the top register. A store operation stores the value from the current top register and then increases ST by 1. Thus, the coprocessor's register stack grows "down" toward lower-addressed registers.

Instructions may address registers either implicitly or explicitly. Instructions that operate at the top of the stack, implicitly address the register pointed to by ST. The instruction, FSQRT, replaces the number at the top with its square root; this instruction takes no operands, because the top-of-stack register is implied as the operand. Other instructions specify the register that is to be used. Explicit register addressing is "top-relative." The expression, ST, denotes the current stack top, and ST(i) refers to the ith register from the ST in the stack. If ST contains "binary 011" (register 3 is at the top of the stack), the instruction, FADD ST,ST(2), would add registers 3 and 5.

Passing subroutine parameters to the register stack eliminates the need for the subroutine to know which registers actually contain the parameters. This allows different routines to call the same subroutine without having to observe a convention for passing parameters in dedicated registers. As long as the stack is not full, each routine simply loads the parameters to the stack and calls the subroutine.



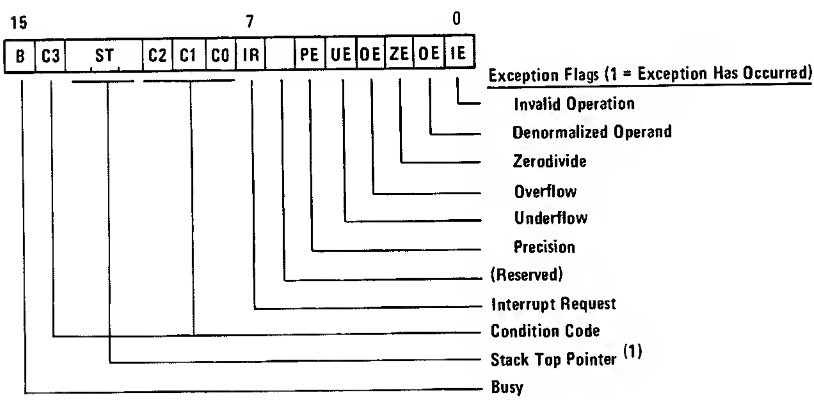
Register Structure

Status Word

The status word reflects the overall condition of the coprocessor. It may be stored in memory with a coprocessor instruction then inspected with a processor code. The status word is divided into the fields shown in the figure below. Bit 15 (BUSY) indicates when the coprocessor is executing an instruction (B=1) or when it is idle (B=0).

Several instructions (for example, the comparison instructions) post their results to the condition code (bits 14 and 10 through 8 of the status word). The main use of the condition code is for conditional branching. This may be accomplished by first executing an instruction that sets the condition code, then storing the status word in memory, and then examining the condition code with processor instructions.

Bits 13 through 11 of the status word point to the coprocessor register that is the current stack top (ST). Bit 7 is the interrupt request field, and bits 5 through 0 are set to indicate that the numeric execution unit has detected an exception while executing the instruction.

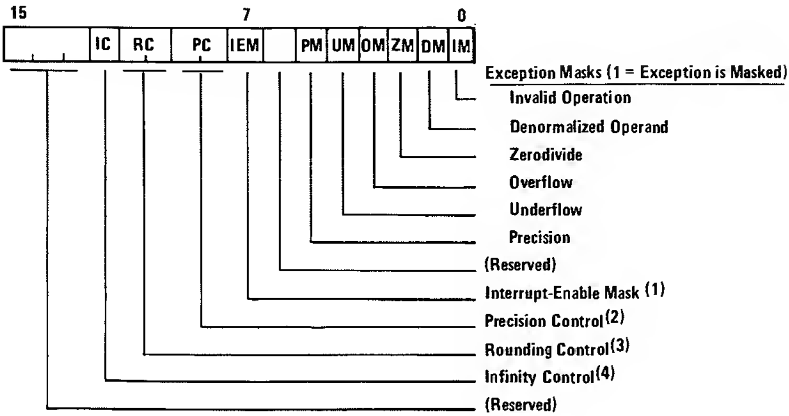


(1) ST values:
000 = register 0 is stack top
001 = register 1 is stack top
.
.
.
111 = register 7 is stack top

Status Word Format

Control Word

The coprocessor provides several options that, are selected by loading a control word register.



(1) Interrupt-Enable Mask:
0 = Interrupts Enabled
1 = Interrupts Disabled (Masked)

(2) Precision Control:
00 = 24 bits
01 = (reserved)
10 = 53 bits
11 = 64 bits

(3) Rounding Control:
00 = Round to Nearest or Even
01 = Round Down (toward ∞)
10 = Round Up (toward ∞)
11 = Chop (Truncate Toward Zero)

(4) Infinity Control:
0 = Projective
1 = Affine

Control Word Format

Tag Word

The tag word marks the content of each register, as shown in the Figure below. The main function of the tag word is to optimize the coprocessor's performance under certain circumstances, and programmers ordinarily need not be concerned with it.

15				7			0
TAG(7)	TAG(6)	TAG(5)	TAG(4)	TAG(3)	TAG(2)	TAG(1)	TAG(0)

Tag values:
- 00 = Valid (Normal or Unnormal)
- 01 = Zero (True)
- 10 = Special (Not-A-Number, ∞, or Denormal)
- 11 = Empty

Tag Word Format

Exception Pointers

The exception pointers in the figure below are provided for user-written exception handlers. When the coprocessor executes an instruction, the control unit saves the instruction address and the instruction opcode in the exception pointer registers. An exception handler subroutine can store these pointers in memory and determine which instruction caused the exception.

OPERAND ADDRESS(1)	
	INSTRUCTION OPCODE(2)
INSTRUCTION ADDRESS(1)	
	10 … 0

(1) 20-bit physical address

(2) 11 least significant bits of opcode: 5 most significant bits are always COPROCESSOR HOOK (11011B)

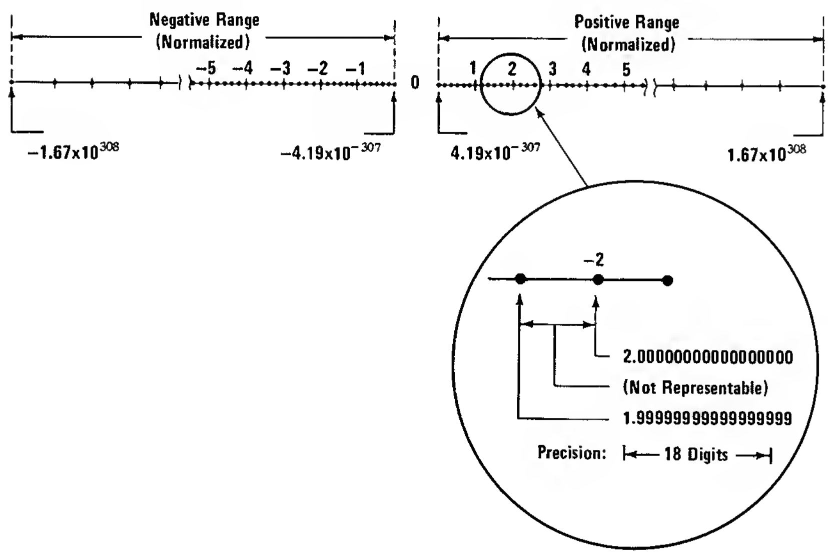
Exception Pointers Format

Number System

The figure below shows the basic coprocessor real number system on a real number line (decimal numbers are shown for clarity, although the coprocessor actually represents numbers in binary). The dots indicate the subset of real numbers the coprocessor can represent as data and final results of calculations. The coprocessor's range is approximately $\pm 4.19 \times 10^{-307}$ to $\pm 1.67 \times 10^{308}$.

The coprocessor can represent a great many of, but not all, the real numbers in its range. There is always a "gap" between two adjacent coprocessor numbers, and the result of a calculation may fall within this space. When this occurs, the coprocessor rounds the true result to a number it can represent.

The coprocessor actually uses a number system that is a superset of that shown in the figure below. The internal format (called temporary real) extends the coprocessor's range to about $\pm 3.4 \times 10^{-4932}$ to $\pm 1.2 \times 10^{4932}$, and its precision to about 19 (equivalent decimal) digits. This format is designed to provide extra range and precision for constants and intermediate results, and is not normally intended for data or final results.



Coprocessor Number System

Instruction Set

On the following pages are descriptions of the operation for the coprocessor's 69 instructions.

An instruction has two basic types of operands – sources and destinations. A source operand simply supplies one of the "inputs" to an instruction; it is not altered by the instruction. A destination operand may also provide an input to an instruction. It is distinguished from a source operand, however, because its content can be altered when it receives the result produced by that operation; that is the destination is replaced by the result.

The operands of any instructions can be coded in more than one way. For example, FADD (add real) may be written without operands, with only a source, or with a destination and a source operand. The instruction descriptions use the simple convention of separating alternative operand forms with slashes; the slashes, however, are not coded. Consecutive slashes indicate there are no explicit operands. The operands for FADD are thus described as:

source/destination, source

This means that FADD may be written in any of three ways:

FADD

FADD source

FADD destination,source

It is important to bear in mind that memory operands may be coded with any of the processor's memory addressing modes.

FABS

FABS (absolute value) changes the top stack element to its absolute value by making its sign positive.

FABS (no operands)			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	14	10-17	0	2	FABS

FADD

Addition

FADD / / source/destination,source

FADDP destination,source

FIADD source

The addition instructions (add real, add real and pop, integer add) add the source and destination operands and return the sum to the destination. The operand at the stack top may be doubled by coding FADD ST,ST(0).

FADD			Exceptions: I, D, O, U, P		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	85	70-100	0	2	FADD ST,ST(4)
short-real	105+EA	90-120+EA	4	2-4	FADD AIR_TEMP [SI]
long-real	110+EA	95-125+EA	8	2-4	FADD [BX],MEAN

FADDP			Exceptions: I, D, O, U, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(I),ST	90	75-105	0	2	FADD ST(2), ST

FIADD			Exceptions: I, D, O, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	120+EA	102-137+EA	2	2-4	FIADD DISTANCE_TRAVELLED
short-integer	125+EA	108-143+EA	4	2-4	FIADD PULSE_COUNT[SI]

FBLD

FBLD Source

FBLD (packed decimal BCD) load)) converts the content of the source operand from packed decimal to temporary real and loads (pushes) the result onto the stack. The packed decimal digits of the source are assumed to be in the range X '0-9H'.

FBLD			Exceptions: I		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
packed-decimal	300+EA	290-310+EA	10	2-4	FBLD YTD_SALES

FBSTP

FBSTP destination

FBSTP (packed decimal (BCD) store and pop) performs the inverse of FBLD, where the stack top is stored to the destination in the packed-decimal data type.

FBSTP			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
packed-decimal	530+EA	520-542+EA	12	2-4	FBSTP [BX].FORCAST

FCHS

FCHS (change sign) complements (reverses) the sign of the top stack element.

FCHS (no operands)			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	15	10-17	0	2	FCHS

FCLEX/FNCLEX

FCLEX/FNCLEX (clear exceptions) clears all exception flags, the interrupt request flag, and the busy flag in the status word.

FCLEX/FNCLEX (no operands)			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FNCLEX

FCOM

FCOM/ /source

FCOM (compare real) compares the stack top to the source operand. This results in the setting of the condition code bits.

FCOM					Exceptions: I, D
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i)	45	40-50	0	2	FCOM ST(1)
short-real	65+EA	63-70+EA	4	2-4	FCOM [BP.] UPPER_LIMIT
long-real	70+EA	65-75+EA	8	2-4	FCOM WAVELENGTH

C3	C0	Order
0	0	ST > source
0	1	ST < source
1	0	ST = source
1	1	ST ? source

NANS and ∞ (projective) cannot be compared and return C3=C0=1 as shown above.

FCOMP

FCOMP/ /source

FCOMP (compare real and pop) operates like FCOM, and in addition pops the stack

FCOMP					Exceptions: I, D
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i)	47	42-52	0	2	FCOMP ST(2)
short-real	68+EA	63-73+EA	4	2-4	FCOMP [8P].N_REAOINGS
long-real	72+EA	67-77+EA	8	2-4	FCOMP DENSITY

FCOMPP

FCOMPP/ /source

FCOMPP (compare real and pop twice) operates like FCOM and, additionally, pops the stack twice, discarding both operands. The comparison is of the stack top to ST(1); no operands may be explicitly coded.

FCOMPP (no operands)			Exceptions: I, D		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	50	45-55	0	2	FCOMPP

FDECSTP

FDECSTP (decrement stack pointer) subtracts 1 from ST, the stack top pointer in the status word.

FDECSTP (no operands)			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	9	6-12	0	2	FOECSTP

FDISI/FNDISI

FDISI/FNDISI (disable interrupts) sets the interrupt enable mask in the control word.

FDISI/FNDISI (no operands)			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FDISI

FDIV

Normal division

FDIV / /source/ destination,source

FDIVP destination,source

FIDIV source

The normal division instructions (divide real, divide real and pop, integer divide) divide the destination by the source and return the quotient to the destination.

FDIV					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i),ST	198	193-203	0	2	FDIV
short-real	220+EA	215-225+EA	4	2-4	FDIV DISTANCE
long-real	225+EA	220-230+EA	8	2-4	FDIV ARC[DI]

FDIVP					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	202	197-207	0	2	FDIVP ST(4), ST

FIDIV					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	230+EA	224-238+EA	2	2-4	FIDIV SURVEY.OBSERVATIONS
short-integer	236+EA	230-243+EA	4	2-4	FIDIV RELATIVE_ANGLE[DI]

FDIVR

Reversed Division

FDIVR / /source/ destination,source

FDIVRP destination,source

FIDIVR source

The reversed division instructions (divide real reversed, divide real reversed and pop, integer divide reversed) divide the source operand by the destination and return the quotient to the destination.

FDIVR					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	199	194-204	0	2	FDIVR ST(2), ST
short-real	221+EA	216-226+EA	6	2-4	FDIVR [BX].PULSE_RATE
long-real	226+EA	221-231+EA	8	2-4	FDIVR RECORDER.FREQUENCY

FDIVRP					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	203	198-208	0	2	FDIVRP ST(1), ST

FIDIVR					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	230+EA	225-239+EA	2	2-4	FIDIVR [BP].X_COORD
short-integer	237+EA	231-245+EA	4	2-4	FIDIVR FREQUENCY

FENI/FNENI

FENI/FNENI (enable interrupts) clear the interrupt enable mask in the control word.

FENI/FNENI (no operands)			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FNENI

FFREE

FFREE destination

FFREE (free register) changes the destination register's tag to empty; the content of the register is not affected.

FFREE			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	186	9-16	0	2	FFREE ST(1)

FICOM

FICOM source

FICOM (integer compare) compares the source to the stack top.

FICOM			Exceptions: I, D		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Exemple
	Typical	Range			
word-integer	80+EA	72-86+EA	2	2-4	FICOM TOOL.N_PASSES
short-integer	85+EA	78-91+EA	2	2-4	FICOM [BP+41].PARM_COUNT

Hardware

FICOMP

FICOMP source

FICOMP (integer compare and pop) operates the same as FICOM and additionally pops the stack.

FICOMP			Exceptions: I, D		
Oparands	Exacution Clocks		Trans-fers 8088	Bytas	Coding Example
	Typical	Range			
word-integer short-inter	82+EA 87+EA	74-88+EA 80-93+EA	2 4	2-4 2-4	FICOMP [BP].LIMIT [SI] FICOMP N_SAMPLES

FILD

FILD source

FILD (integer load) loads (pushes) the source onto the stack.

FILD			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
word-integer short-integer long-integer	50+EA 56+EA 64+EA	46-54+EA 52-60+EA 60-68+EA	2 4 8	2-4 2-4 2-4	FILD [BX].SEQUENCE FILD STANDOFF[DI] FILD RESPONSE.COUNT

FINCSTP

FINCSTP (increment stack pointer) adds 1 to the stack top pointer (ST) in the status word.

FINCSTP (no operands)			Exceptions: None		
Oparands	Execution Clocks		Trans-fars 8088	Bytas	Coding Exampla
	Typical	Range			
(no oparands)	9	6-12	0	2	FINCSTP

FINIT/FNINIT

FINIT/FNINIT (initialize processor) performs the functional equivalent of a hardware RESET.

FINIT/FNINIT (no operands)			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FINIT

Field	Value	Interpretation
Control Word		
Infinity Control	0	Projective
Rounding Control	00	Round to nearest
Precision Control	11	64 bits
Interrupt-enable Mask	1	Interrupts disabled
Exception Masks	111111	All exceptions masked
Status Word		
Busy	0	Not Busy
Condition Code	????	(Indeterminate)
Stack Top	000	Empty stack
Interrupt Request	0	No interrupt
Exception Flags	000000	No exceptions
Tag Word		
Tags	11	Empty
Registers	N.C.	Not changed
Exception Pointers		
Instruction Code	N.C.	Not changed
Instruction Address	N.C.	Not changed
Operand Address	N.C.	Not changed

FIST

FIST destination

FIST (integer store) stores the stack top to the destination in the integer format.

FIST			Exceptions: I, P		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	86+EA	80-90+EA	4	2-4	FIST OBS.COUNT[SI]
short-integer	88+EA	82-92+EA	6	2-4	FIST [BP].FACTORED_PULSES

FISTP

FISTP destination

FISTP (integer store and pop) operates like FIST and also pops the stack following the transfer. The destination may be any of the binary integer data types.

FISTP			Exceptions: I, P		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	88+EA	82-92+EA	4	2-4	FISTP [BX].ALPHA_COUNT[SI]
short-integer	90+EA	84-94+EA	6	2-4	FISTP CORRECTED_TIME
long-integer	100+EA	94-105+EA	10	2-4	FISTP PANEL.N_READINGS

FLD

FLD source

FLD (load real) loads (pushes) the source operand onto the top of the register stack.

FLD			Exceptions: I, D		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	20	17-22	0	2	FLD ST(0)
short-real	43+EA	38-56+EA	4	2-4	FLD READING[SI].PRESSURE
long-real	46+EA	40-60+EA	8	2-4	FLD [BP].TEMPERATURE
temp-real	57+EA	53-65+EA	10	2-4	FLD SAVEREADING

FLDCW

FLDCW source

FLDCW (load control word) replaces the current processor control word with the word defined by the source operand.

FLDCW			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
2-bytes	10+EA	7-14+EA	2	2-4	FLDCW CONTROL_WORD

FLDENV

FLDENV source

FLDENV (load environment) reloads the coprocessor environment from the memory area defined by the source operand.

FLDENV			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Exemple
	Typicel	Range			
14-bytes	40+EA	35-45+EA	14	2-4	FLDENV [BP+6]

FLDLG2

FLDLG2 (load log base 10 of 2) loads (pushes) the value of $LOG_{10}2$ onto the stack.

FLDLG2 (no operands)			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Renge			
(no operands)	21	18-24	0	2	FLDLG2

FLDLN2

FLDLN2 (load log base e of 2) loads (pushes) the value of LOG_e2 onto the stack.

FLDLN2 (no operands)			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	20	17-23	0	2	FLDLN2

FLDL2E

FLDL2E (load log base 2 of e) loads (pushes) the value LOG_2e onto the stack.

FLDL2E (no operands)			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	18	15-21	0	2	FLDL2E

FLDL2T

FLDL2T (load log base 2 of 10) loads (pushes) the value of $LOG_2 10$ onto the stack.

FLDL2T (no operands)			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	19	16-22	0	2	FLDL2T

FLDPI

FLDPI (load π) loads (pushes) π onto the stack.

FLDPI (no operands)			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	8ytes	Coding Example
	Typical	Range			
(no operands)	19	16-22	0	2	FLDPI

FLDZ

FLDZ (load zero) loads (pushes) +0.0 onto the stack.

FLDZ (no operands)			Exceptions: I		
Operends	Execution Clocks		Trans-fers 8088	Bytes	Coding Exampla
	Typical	Range			
(no operands)	14	11-17	0	2	FLD1

FLD1

FLD1 (load one) loads (pushes) +1.0 onto the stack.

FLD1 (no operands)			Exceptions: I		
Oparends	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	18	15-21	0	2	FLDZ

FMUL

Multiplication

FMUL / /source/destination,source

FMULP destination,source

FIMUL source

The multiplication instructions (multiply real, multiply real and pop, integer multiply) multiply the source and destination operands and return the product to the destination. Coding FMUL ST,ST(0) square the content of the stack top.

FMUL			Exceptions: I, D, O, U, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i),ST/ST,ST(i)[1]	97	90-105	0	2	FMUL ST,ST(3)
//ST(i),ST/ST,ST(i)	138	130-145	0	2	FMUL ST,ST(3)
short-real	118+EA	110-125+EA	4	2-4	FMUL SPEED_FACTOR
long-real[1]	120+EA	112-126+EA	8	2-4	FMUL [BP].HEIGHT
long-real	161+EA	154-168+EA	8	2-4	FMUL [BP].HEIGHT

[1] occurs when one or both operands is "short" - it has 40 trailing zeros in its fraction.

FMULP			Exceptions: I, D, O, U, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST[1]	100	94-108	0	2	FMULP ST(1),ST
ST(i),ST	142	134-148	0	2	FMULP ST(1),ST

[1] occurs when one or both operands is "short" - it has 40 trailing zeros in its fraction.

FIMUL			Exceptions: I, D, O, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	130+EA	124-138+EA	2	2-4	FIMUL BEARING
short-integer	136+EA	130-144+EA	4	2-4	FIMUL POSITION.Z_AXIS

FNOP

FNOP (no operation) stores the stack top to the stack top (FST ST,ST(0)) and thus effectively performs no operation.

FNOP (no operands)			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	13	10-16	0	2	FNOP

FPATAN

FPATAN (partial arctangent) computes the function θ =ARCTAN (Y/X). X is taken from the top stack element and Y from ST(1). Y and X must observe the inequality $0<Y<X<\infty$. The instruction pops the stack and returns θ to the (new) stack top, overwriting the Y operand.

FPATAN (no operands)			Exceptions: U, P (operands not checked)		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	650	250-800	0	2	FPATAN

FPREM

FPREM (partial remainder) performs modulo division on the top stack element by the next stack element, that is, ST(1) is the modulus.

FPREM (no operands)			Exceptions: I, D, U		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Exemple
	Typical	Range			
(no operands)	125	15-190	0	2	FPREM

FPTAN

FPTAN (partial tangent) computes the function Y/X =TAN(θ). θ is taken from the top stack element; it must lie in the range $0<\theta<\pi/4$. The result of the operation is a ratio; Y replaces θ in the stack and X is pushed, becoming the new stack top.

FPTAN			Exceptions: I, P (operands not checked)		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	450	30-540	0	2	FPTAN

FRNDINT

FRNDINT (round to integer) rounds the top stack element to an integer.

FRNDINT (no operands)			Exceptions: I, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	45	16-50	0	2	FRNDINT

FRSTOR

FRSTOR source

FRSTOR (restore state) reloads the coprocessor from the 94-byte memory area defined by the source operand.

FRSTOR			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
94-bytes	210+EA	205-215+EA	96	2-4	FRSTOR [BP]

FSAVE/FNSAVE

FSAVE/FNSAVE destination

FSAVE/FNSAVE (save state) writes the full coprocessor state – environment plus register stack – to the memory location defined by the destination operand.

FSAVE/FNSAVE			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
94-bytes	210+EA	205-215+EA	94	2-4	FSAVE [BP]

FSCALE

FSCALE (scale) interprets the value contained in ST(1) as an integer, and adds this value to the exponent of the number in ST. This is equivalent to:

$$ST \leftarrow ST \bullet 2^{ST(1)}$$

Thus, FSCALE provides rapid multiplication or division by integral powers of 2.

FSCALE (no operands)			Exceptions: I, O, U		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	35	32-38	0	2	FSCALE

FSQRT

FSQRT (square root) replaces the content of the top stack element with its square root.

Note: the square root of −0 is defined to be −0.

FSQRT (no operands)			Exceptions: I, D, P		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	183	180-186	0	2	FSQRT

FST

FST destination

FST (store real) transfers the stack top to the destination, which may be another register on the stack or long real memory operand.

FST			Exceptions: I, O, U, P		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	18	15-22	0	2	FST ST(3)
short-real	87+EA	84-90+EA	6	2-4	FST CORRELATION [DI]
long-real	100+EA	96-104+EA	10	2-4	FST MEAN_READING

FSTCW/FNSTCW

FSTCW/FNSTCW destination

FSTCW/FNSTCW (store control word) writes the current processor control word to the memory location defined by the destination.

FSTCW/FNSTCW			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
2-bytes	15+EA	12-18+EA	4	2-4	FSTCW SAVE_CONTROL

FSTENV/FNSTENV

FSTENV/FNSTENV destination

FSTENV/FNSTENV (store environment) writes the coprocessor's basic status – control, status and tag words, and exception pointers – to the memory location defined by the destination operand.

FSTENV/FNSTENV			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
14-bytes	45+EA	40-50+EA	16	2-4	FSTENV [BP]

FSTP

FSTP destination

FSTP (store real and pop) operates the same as FST, except that the stack is popped following the transfer.

FSTP			Exceptions: I, O, U, P		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	20	17-24	0	2	FSTP ST(2)
short-real	89+EA	86-92+EA	6	2-4	FSTP [BX].ADJUSTED_RPM
long-real	102+EA	98-106+EA	10	2-4	FSTP TOTAL_DOSAGE
temp-real	55+EA	52-58+EA	12	2-4	FSTP REG_SAVE[SI]

FSTSW/FNSTSW

FSTSW/FNSTSW destination

FSTSW/FNSTSW (store status word) writes the current value of the coprocessor status word to the destination operand in memory.

FSTSW/FNSTSW			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
2-bytes	14+EA	12-18+EA	4	2-4	FSTSW SAVE_STATUS

FSUB

Subtraction

FSUB / /source/destination,source

FSUBP destination,source

FISUB source

The normal subtraction instructions (subtract real, subtract real and pop, integer subtract) subtract the source operand from the destination and return the difference to the destination.

FSUB			Exceptions: I, D, O, U, P		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	85	70-100	0	2	FSUB ST,ST(2)
short-real	105+EA	90-120+EA	4	2-4	FSUB BASE_VALUE
long-real	110+EA	95-125+EA	8	2-4	FSUB COORDINATE.X

FSUBP					Exceptions: I, D, O, U, P
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	90	75-105	0	2	FSUBP ST(2),ST

FISUB					Exceptions: I, D, O, P
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	120+EA	102-137+EA	2	2-4	FISUB BASE_FREQUENCY
short-integer	125+EA	108-143+EA	4	2-4	FISUB TRAIN_SIZE[DI]

FSUBR

Reversed Subtraction

FSUBR / /source/destination,source

FSUBRP destination,source

FISUBR source

The reversed subtraction instructions (subtract real reversed, subtract real reversed and pop, integer subtract reversed) subtract the destination from the source and return the difference to the destination.

FSUBR					Exceptions: I, D, O, U, P
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	87	70-100	0	2	FSUBR ST,ST(1)
short-real	105+EA	90-120+EA	4	2-4	FSUBR VECTOR[SI]
long-real	110+EA	95-125+EA	8	2-4	FSUBR [BX].INDEX

FSUBRP	Exceptions: I, D, O, U, P				
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	90	75-105	0	2	FSUBRP ST(1),ST

FISUBR	Exceptions: I, D, O, P				
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	120+EA	103-139+EA	2	2-4	FISUBR FLOOR[BX] [SI]
short-integer	125+EA	109-144+EA	4	2-4	FISUBR BALANCE

FTST

FTST (test) tests the top stack element by comparing it to zero. The result is posted to the condition codes.

FTST (no operands)			Exceptions: I, D		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	42	38-48	0	2	FTST

C3	C0	Result
0	0	ST is positive and nonzero
0	1	ST is negative and nonzero
1	0	ST is zero (+ or –)
1	1	ST is not comparable (that is, it is a NAN or projective ∞)

FWAIT

FWAIT (processor instruction)

FWAIT is not actually a coprocessor instruction, but an alternate mnemonic for the processor WAIT instruction. The FWAIT mnemonic should be coded whenever the programmer wants to synchronize the processor to the coprocessor, that is, to suspend further instruction decoding until the coprocessor has completed the current instruction.

FWAIT (no operands)			Exceptions: Non (CPU instruction)		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Exemple
	Typical	Range			
(no operands)	3+5n	3+5n	0	1	FWAIT

FXAM

FXAM (examine) reports the content of the top stack element as positive/negative and NAN/unnormal/denormal/normal/zero, or empty.

FXAM			Exceptions: None		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Exemple
	Typical	Renge			
(no operands)	17	12-23	0	2	FXAM

Condition Code				Interpretation
C3	C2	C1	C0	
0	0	0	0	+ Unnormal
0	0	0	1	+ NAN
0	0	1	0	– Unnormal
0	0	1	1	– NAN
0	1	0	0	+ Normal
0	1	0	1	+ ∞
0	1	1	0	– Normal
0	1	1	1	– ∞
1	0	0	0	+ 0
1	0	0	1	Empty
1	0	1	0	– 0
1	0	1	1	Empty
1	1	0	0	+ Denormal
1	1	0	1	Empty
1	1	1	0	– Denormal
1	1	1	1	Empty

FXCH

FXCH/ /destination

FXCH (exchange registers) swaps the contents of the destination and the stack top registers. If the destination is not coded explicitly, ST(1) is used.

FXCH			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i)	12	10-15	0	2	FXCH ST(2)

FXTRACT

FXTRACT (extract exponent and significant) "decomposes" the number in the stack top into two numbers that represent the actual value of the operand's exponent and significand fields contained in the stack top and ST(1).

FXTRACT			Exceptions: I		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	50	27-55	0	2	FXTRACT

FYL2X

FYL2X (Y log base 2 of X) calculates the function $Z=Y \bullet LOG_2$. X is taken from the stack top and Y from ST(1). The operands must be in the ranges $0<X<\infty$ and $-\infty<Y<+\infty$. The instruction pops the stack and returns Z at the (new) stack top, replacing the Y operand.

$$LOG_n 2 \bullet LOG_2 X$$

FYL2X			Exceptions: P (operands not checked)		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	950	900-1100	0	2	FYL2X

FYL2XP1

FYL2XP1 (Y log base 2 of (X + 1)) calculates the function $Z = Y \bullet LOG_2(X+1)$. X is taken from the stack top and must be in the range $0 < |X| < (1-(\sqrt{2}/2))$. Y is taken from ST(1) and must be in the range $-\infty < Y < \infty$. FYL2XP1 pops the stack and returns Z at the (new) stack top, replacing Y.

FYL2XP1			Exceptions: P (operands not checked)		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	850	700-1000	0	2	FYL2XP1

F2XM1

F2XM1 (2 to the X minus 1) calculates the function $Y=2^x-1$. X is taken from the stack top and must be in the range $0<X<0.5$. The result Y replaces the stack top.

This instruction is designed to produce a very accurate result even when X is close to zero. To obtain $Y=2^x$, add 1 to the result delivered by F2XM1.

F2XM1			Exceptions: U, P (operands not checked)		
Operands	Execution Clocks		Trans-fers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	500	310-630	0	2	F2XM1

Notes:

IBM Keyboard

The keyboard has a permanently attached cable that connects to a DIN connector at the rear of the system unit. This shielded four-wire cable has power (+5 Vdc), ground, and two bidirectional signal lines. The cable is approximately 6-feet long and is coiled, like that of a telephone handset.

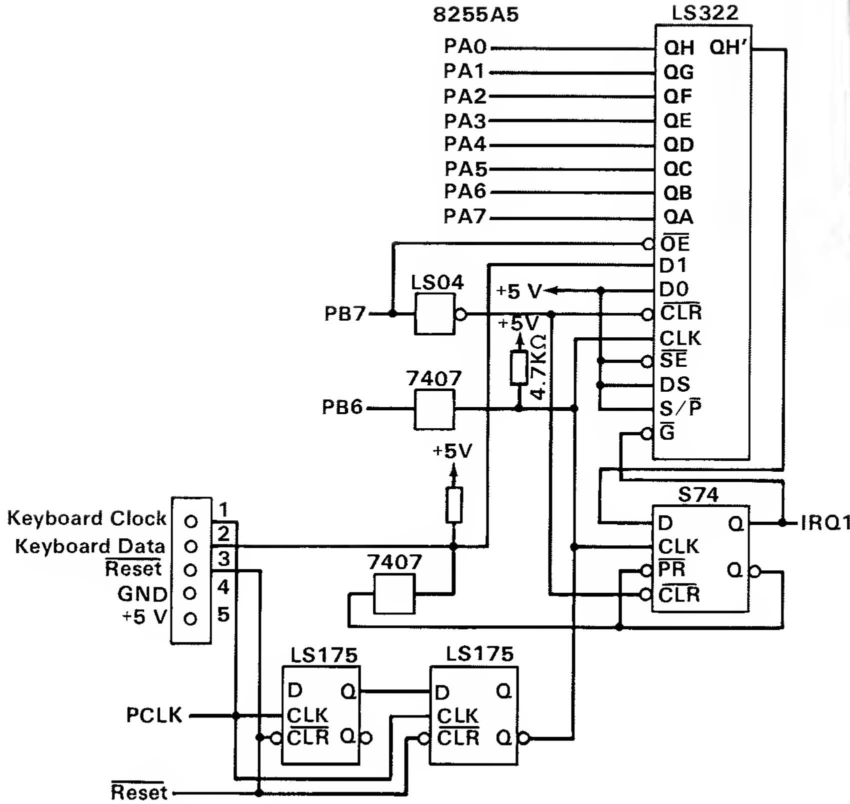
The keyboard uses a capacitive technology with a microcomputer (Intel 8048) performing the keyboard scan function. The keyboard has three tilt positions for operator comfort (5-, 7-, or 15-degree tilt orientations).

The keyboard has 83 keys arranged in three major groupings. The central portion of the keyboard is a standard typewriter keyboard layout. On the left side are 10 function keys. These keys are user-defined by the software. On the right is a 15-key keypad. These keys are also defined by the software, but have legends for the functions of numeric entry, cursor control, calculator pad, and screen edit.

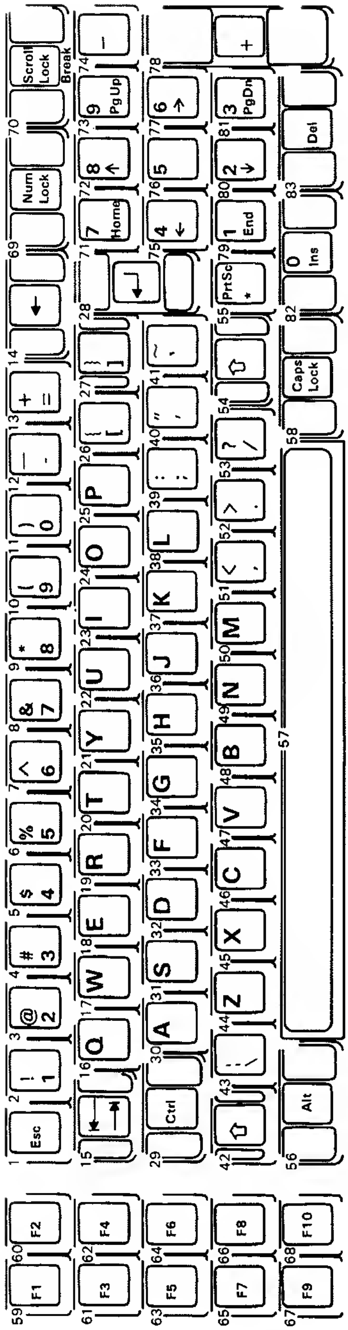
The keyboard interface is defined so that system software has maximum flexibility in defining certain keyboard operations. This is accomplished by having the keyboard return scan codes rather than American Standard Code for Information Interchange (ASCII) codes. In addition, all keys are typematic and generate both a make and a break scan code. For example, key 1 produces scan code hex 01 on make and code hex 81 on break. Break codes are formed by adding hex 80 to make codes. The keyboard I/O driver can define keyboard keys as shift keys or typematic, as required by the application.

The microcomputer (Intel 8048) in the keyboard performs several functions, including a power-on self-test when requested by the system unit. This test checks the microcomputer ROM, tests memory, and checks for stuck keys. Additional functions are: keyboard scanning, buffering of up to 16 key scan codes, maintaining bidirectional serial communications with the system unit, and executing the hand-shake protocol required by each scan-code transfer.

The following pages have figures that show the keyboard, the scan codes, and the keyboard interface connector specifications.



Keyboard Interface Block Diagram

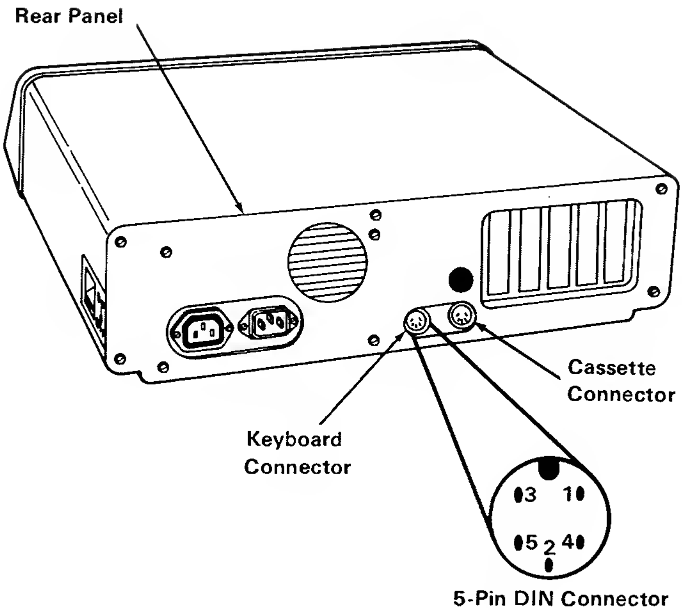


Note: Nomenclature is on both the top and front face of the keybutton as shown. The number to the upper left designates the button position.

Keyboard Diagram

Key Position	Scan Code in Hex	Key Position	Scan Code in Hex
1	01	43	2B
2	02	44	2C
3	03	45	2D
4	04	46	2E
5	05	47	2F
6	06	48	30
7	07	49	31
8	08	50	32
9	09	51	33
10	0A	52	34
11	0B	53	35
12	0C	54	36
13	0D	55	37
14	0E	56	38
15	0F	57	39
16	10	58	3A
17	11	59	3B
18	12	60	3C
19	13	61	3D
20	14	62	3E
21	15	63	3F
22	16	64	40
23	17	65	41
24	18	66	42
25	19	67	43
26	1A	68	44
27	1B	69	45
28	1C	70	46
29	1D	71	47
30	1E	72	48
31	1F	73	49
32	20	74	4A
33	21	75	4B
34	22	76	4C
35	23	77	4D
36	24	78	4E
37	25	79	4F
38	26	80	50
39	27	81	51
40	28	82	52
41	29	83	53
42	2A		

Keyboard Scan Codes



Pin	TTL Signal	Signal Level
1	+Keyboard Clock	+5 Vdc
2	+Keyboard Data	+5 Vdc
3	–Keyboard Reset (Not used by keyboard)	
	Power Supply Voltages	**Voltage**
4	Ground	0
5	+5 Volts	+5 Vdc

Keyboard Interface Connector Specifications

Expansion Unit

The expansion unit option upgrades the IBM Personal Computer by adding expansion slots in a separate unit. This option consists of an extender card, an expansion cable, and the expansion unit. The expansion unit contains a power supply, an expansion board, and a receiver card. This option utilizes one expansion slot in the system unit to provide seven additional expansion slots in the expansion unit.

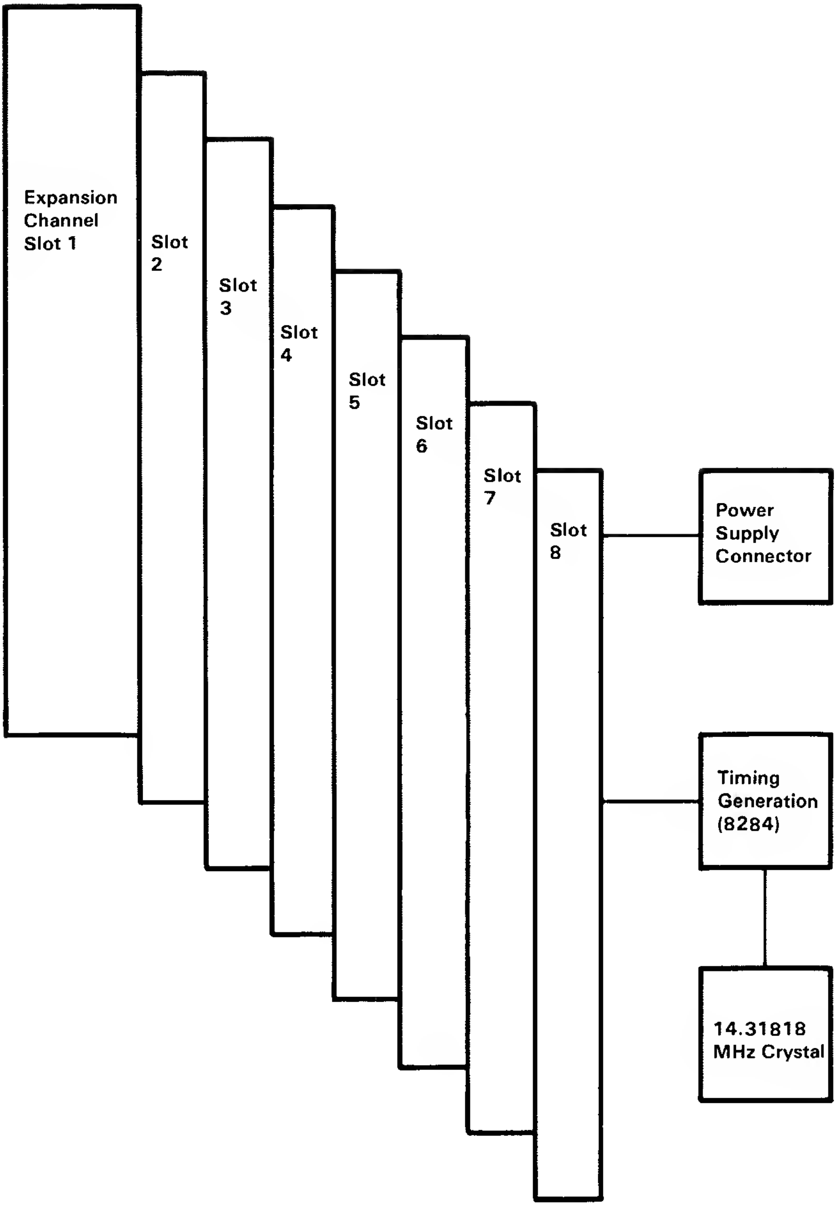
Expansion Unit Cable

The expansion unit cable consists of a 56-wire, foil-shielded cable terminated on each end with a 62-pin D-shell male connector. Either end of the expansion unit cable can be plugged into the extender card or the receiver card.

Expansion Board

The expansion board is a support board that carries the I/O channel signals from the option adapters and receiver card. These signals, except 'osc,' are carried over the expansion cable. Because 'osc' is not sent over the expansion cable, a 14.31818-MHz signal is generated on the expansion board. This signal may not be in phase with the 'osc' signal in the system unit.

Decoupling capacitors provided on the expansion board aid in noise filtering.



Expansion Board Block Diagram

Expansion Channel

All signals found on the system unit's I/O channel will be provided to expansion slots in the expansion unit, with the exception of the 'osc' signal and the voltages mentioned previously.

A 'ready' line on the expansion channel makes it possible to operate with slow I/O or memory devices. If the channel's 'I/O ch rdy' line is not activated by an addressed device, all processor-generated memory cycles take five processor clock cycles per byte for memory in the expansion unit.

The following table contains a list of all the signals that are redriven by the extender and receiver cards, and their associated time delays. The delay times include the delay due to signal propagation in the expansion cable. Assume a nominal cable delay of 3 ns. As such, device access will be less than 260 ns.

Signal	Nominal Delay (ns)	Maximum Delay (ns)	Direction (*)
A0 - A19	27	39	Output
AEN	27	39	Output
DACK0 - DACK3	27	39	Output
MEMR	27	39	Output
MEMW	51	75	Output
IOR	51	75	Output
IOW	27	39	Output
ALE	27	39	Output
CLK	27	39	Output
T/C	27	39	Output
RESET	27	39	Output
IRQ2 - IRQ7	36	(**)	Input
DRQ1 - DRQ3	36	(**)	Input
I/O CH RDY	36	51	Input
I/O CH CK	36	51	Input
D0 - D7 (Read)	84	133	Input
D0 - D7 (Write)	19	27	Output

(*) With respect to the system unit.

(**) Asynchronous nature of interrupts and other requests are more dependent on processor recognition than electrical signal propagation through expansion logic.

Power Supply

The expansion unit dc power supply is a 130-watt, 4 voltage level switching regulator. It is integrated into the expansion unit and supplies power for the expansion unit, and its options. The supply provides 15 A of +5 Vdc, plus or minus 5%, 4.2A of +12 Vdc, plus or minus 5%, 300 mA of −5 Vdc, plus or minus 10%, and 250 mA of −12 Vdc, plus or minus 10%. All power levels are regulated with over-voltage and over-current protection. The input is 120 Vac and fused. If dc over-load or over-voltage conditions exist, the supply automatically shuts down until the condition is corrected. The supply is designed for continuous operation at 130 watts.

The power supply is located at the right rear of the expansion unit. It supplies operating voltages to the expansion board, and provides two separate connections for power to the fixed disk drives. The nominal power requirements and output voltages are listed in the following tables:

Voltage (Vac at 50/60 Hz)			Frequency (Hz)	Current (Amps)
Nominal	Minimum	Maximum	+/- 3 Hz	Maximum
110	90	137	50/60	4.1 at 90 Vac

Input Requirements

Voltage (Vdc)	Current (Amps)		Regulation (Tolerance)	
Nominal	Minimum	Maximum	+%	-%
+5.0	2.3	15.0	5	4
−5.0	0.0	0.3	10	8
+12.0	0.4	4.2	5	4
−12.0	0.0	0.25	10	9

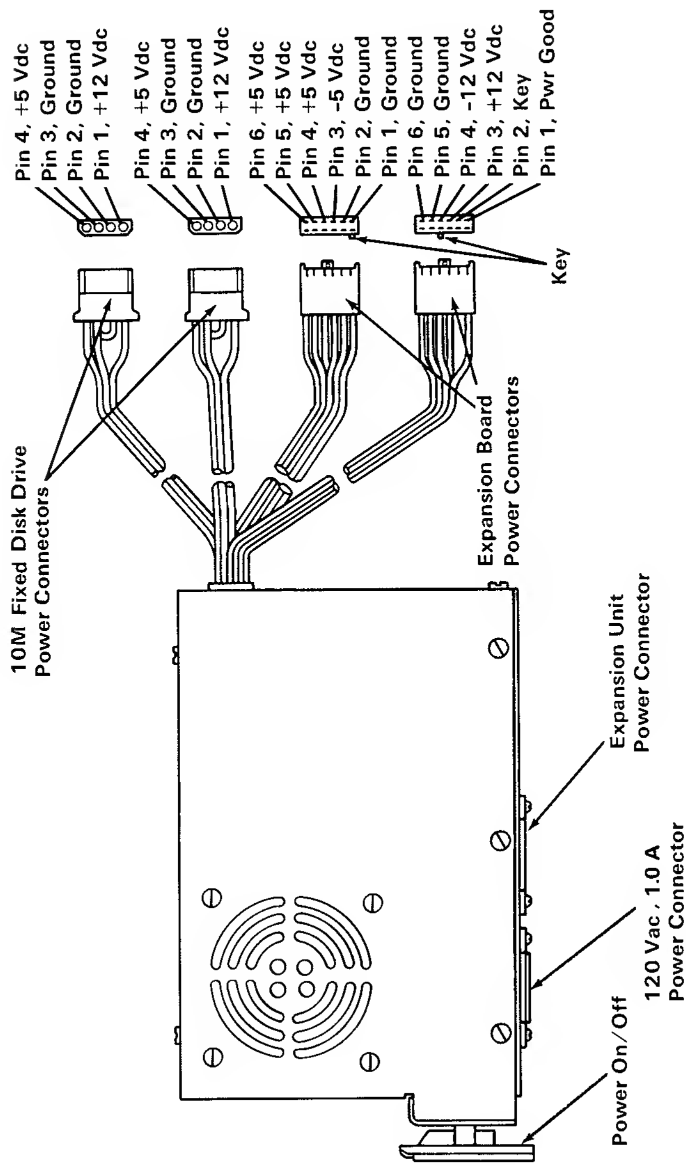
Vdc Output

Voltage (Vac)	Current (Amps)		Voltage Limits (Vac)	
Nominal	Minimum	Maximum	Minimum	Maximum
120	0.0	1.0	88	137

Vac Output

Power Supply Connectors and Pin Assignments

The power connector on the expansion board is a 12-pin male connector that plugs into the power-supply connectors. The pin configurations and locations are shown below:



Power Supply and Connectors

Over-Voltage/Over-Current Protection

Voltage Nominal Vac	Type Protection	Rating Amps
110	Fuse	5

Power On/Off Cycle: When the supply is turned off for a minimum of 1.0 second, and then turned on, the power-good signal will be regenerated.

The power-good signal indicates that there is adequate power to continue processing. If the power goes below the specified levels, the power-good signal triggers a system shutdown.

This signal is the logical AND of the dc output-voltage sense signal and the ac input voltage fail signal. This signal is TTL-compatible up-level for normal operation or down-level for fault conditions. The ac fail signal causes power-good to go to a down-level when any output voltage falls below the regulation limits.

The dc output-voltage sense signal holds the power-good signal at a down level (during power-on) until all output voltages have reached their respective minimum sense levels. The power-good signal has a turn-on delay of at least 100 ms but no greater than 500 ms.

The sense levels of the dc outputs are:

Output (Vdc)	Minimum (Vdc)	Sense Voltage Nominal (Vdc)	Maximum (Vdc)
+5	+4.5	+5.0	+5.5
−5	-4.3	−5.0	-5.5
+12	+10.8	+12.0	+13.2
−12	-10.2	−12.0	-13.2

Extender Card

The extender card is a four-plane card. The extender card redrives the I/O channel to provide sufficient power to avoid capacitive effects of the cable. The extender card presents only one load per line of the I/O channel.

The extender card has a wait-state generator that inserts a wait-state on 'memory read' and 'memory write' operations (except refreshing) for all memory contained in the expansion unit. The address range for wait-state generation is controlled by switch settings on the extender card.

The DIP switch on the extender card should be set to indicate the maximum contiguous read/write memory housed in the system unit. The extender card switch settings are located in "Appendix G: Switch Settings." Switch positions 1 through 4 correspond to address bits hex A19 to hex A16, respectively.

The switch settings determine which address segments have a wait state inserted during 'memory read' and 'memory write' operations. Wait states are required for any memory, including ROM on option adapters, in the expansion unit. Wait states are not inserted in the highest segment, hex addresses F0000 to FFFFF (segment F).

Extender Card Programming Considerations

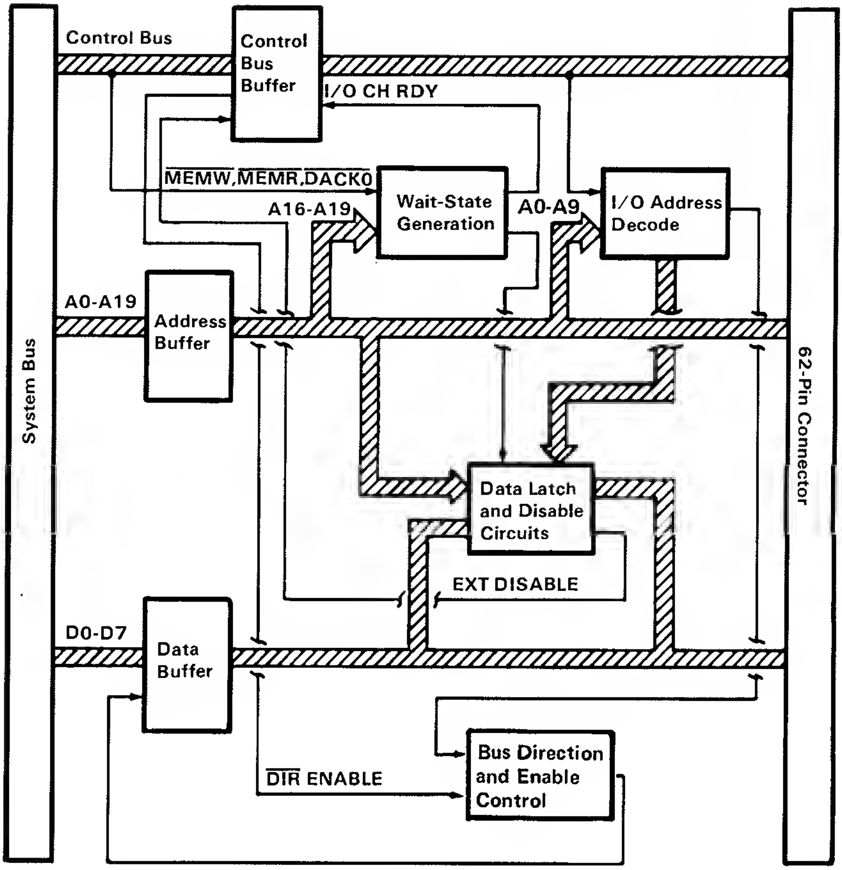
Several registers associated with the expansion option are programmable and readable for diagnostic purposes. The following figure indicates the locations and functions of the registers on the extender card.

Location	Function
Memory FXXXX(*)	Write to memory to latch address bits
Port 210	Write to latch expansion bus data (ED0 - ED7)
Port 210	Read to verify expansion bus data (ED0 - ED7)
Port 211	Read high-order address bits (A8 - A15)
Port 211	Write to clear wait test latch
Port 212	Read low-order address bits (A0 - A7)
Port 213	Write 00 to disable expansion unit
Port 213	Write 01 to enable expansion unit
Port 213	Read status of expansion unit D0 = enable/disable D1 = wait-state request flag D2-D3 = not used D4-D7 = switch position 1 = Off 0 = On

(*) Example: Write to memory location F123:4=00
Read Port 211 = 12
Read Port 212 = 34

(All values in hex)

The expansion unit is automatically enabled upon power-up. The extender card and receiver card will both be written to, if the expansion unit is not disabled when writing to FXXXX. However, the system unit and the expansion unit are read back separately.



Extender Card Block Diagram

Receiver Card

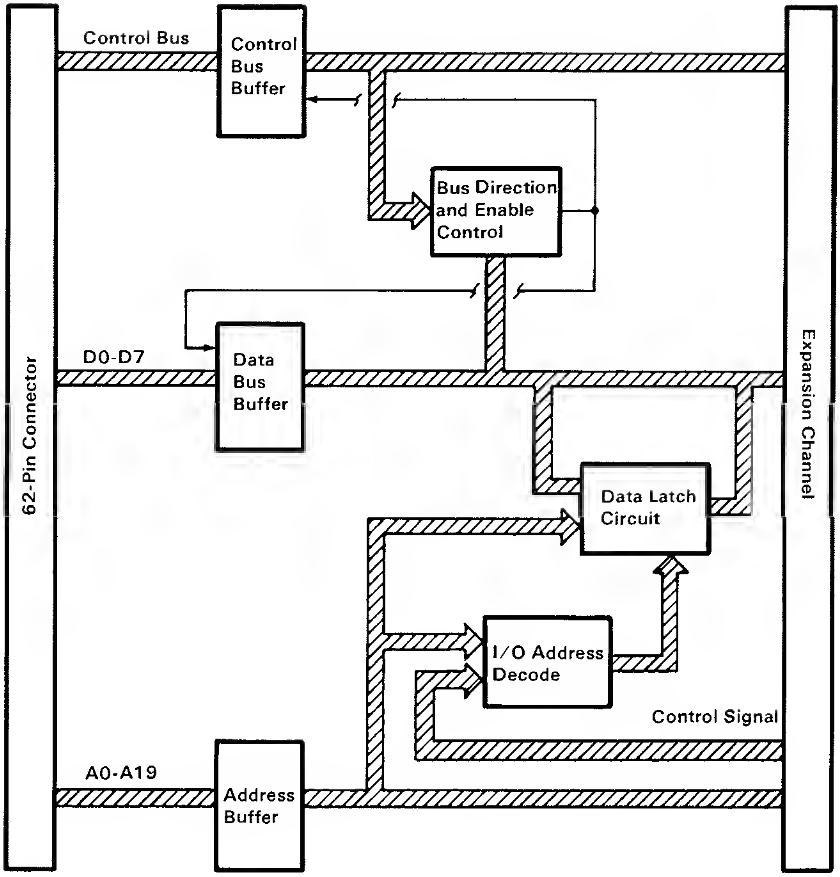
The receiver card is a four-plane card that fits in expansion slot 8 of the expansion unit. The receiver card redrives the I/O channel to provide sufficient power for additional options and to avoid capacitive effects. Directional control logic is contained on the receiver card to resolve contention and direct data flow on the I/O channel. Steering signals are transmitted back over the expansion cable for use on the extender card.

Receiver Card Programming Considerations

Several registers associated with the expansion option are programmable and readable for diagnostic purposes. The following figure indicates the locations and functions of the registers on the receiver card.

Location	Function
Memory FXXXX(*)	Write to memory to latch address bits
Port 214	Write to latch data bus bits (D0 - D7)
Port 214	Read data bus bits (D0 - D7)
Port 215	Read high-order address bits (A8 - A15)
Port 216	Read low-order address bits (A0 - A7)
(*) Example:	Write to memory location F123:4=00 Read Port 215 =12 Read Port 216 =34
(All values in hex)	

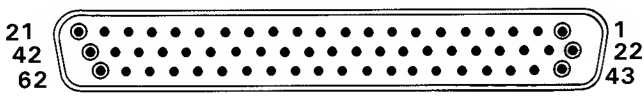
The expansion unit is automatically enabled upon power-up. The expansion unit and the system unit will be written to, if the expansion unit is not disabled when writing to FXXXX. However, the system unit and the expansion unit are read back separately.



Receiver Card Block Diagram

Expansion Unit Interface Information

The extender card and receiver card rear-panel connectors are the same. Pin and signal assignments for the extender and receiver cards are shown below.



Pin	Signal	Pin	Signal	Pin	Signal
1	+E IRQ6	22	+E D5	43	+E IRQ7
2	+E DRQ2	23	+E DRQ1	44	+E D6
3	+E DIR	24	+E DRQ3	45	+E I/O CH RDY
4	+E ENABLE	25	RESERVED	46	+E IRQ3
5	+E CLK	26	+E ALE	47	+E D7
6	-E MEM IN EXP	27	+E T/C	48	+E D1
7	+E A17	28	+E RESET	49	-E I/O CH CK
8	+E A16	29	+E AEN	50	+E IRQ2
9	+E A5	30	+E A19	51	+E D0
10	-E DACK0	31	+E A14	52	+E D2
11	+E A15	32	+E A12	53	+E D4
12	+E A11	33	+E A18	54	+E IRQ5
13	+E A10	34	-E MEMR	55	+E IRQ4
14	+E A9	35	-E MEMW	56	+E D3
15	+E A1	36	+E A0	57	GND
16	+E A3	37	-E DACK3	58	GND
17	-E DACK1	38	+E A6	59	GND
18	+E A4	39	-E IOR	60	GND
19	-E DACK2	40	+E A8	61	GND
20	-E IOW	41	+E A2	62	GND
21	+E A13	42	+E A7		

E = Extended

Connector Specifications

IBM 80 CPS Printers

The IBM 80 CPS (characters-per-second) Printers are self-powered, stand-alone, tabletop units. They attach to the system unit through a parallel signal cable, 6 feet in length. The units obtain ac power from a standard wall outlet (120 Vac). The printers are 80 cps, bidirectional, wire-matrix devices. They print characters in a 9 by 9 dot matrix with a 9-wire head. They can print in a compressed mode of 132 characters per line, in a standard mode of 80 characters per line, in a double width, compressed mode of 66 characters per line, and in a double width mode of 40 characters per line. The printers can print double-size characters and double-strike characters. The printers print the standard ASCII, 96-character, uppercase and lowercase character sets. A printer without an extended character set also has a set of 64 special block graphic characters.

The IBM 80 CPS Graphics Printer has additional capabilities including: an extended character set for international languages, subscript, superscript, an underline mode, and programmable graphics.

The printers can also accept commands setting the line-feed control desired for the application. They attach to the system unit through the printer adapter or the combination monochrome display and printer adapter. The cable is a 25-lead shielded cable with a 25-pin D-shell connector at the system unit end, and a 36-pin connector at the printer end.

(1)	**Print Method:**	Serial-impact dox matrix	
(2)	**Print Speed:**	80 cps	
(3)	**Print Direction:**	Bidirectional with logical seeking	
(4)	**Number of Pins in Head:**	9	
(5)	**Line Spacing:**	1/16 inch (4.23 mm) or programmable	
(6)	**Printing Characteristics**		
	Matrix:	9 x 9	
	Character Set:	Full 96-character ASCII with descenders plus 9 international characters/symbols.	
	Graphic Character:	See "Additional Printer Specifications"	
(7)	**Printing Sizes**		
		Characters per inch	Maximum characters per inch
	Normal:	10	80
	Double Width:	5	40
	Compressed:	16.5	132
	Double Width-Compressed:	8.25	66
(8)	**Media Handling**		
	Paper Feed:	Adjustable sprocket pin feed	
	Paper Width Range:	4 inch (101.6 mm) to 10 inch (254 mm)	
	Copies:	One original plus two carbon copies (total thickness not to exceed 0.012 inch (0.3 mm)). Minimum paper thickness is 0.0025 inch (0.064 mm).	
	Paper Path:	Rear	
(9)	**Interfaces**		
	Standard:	Parallel 8-bit Data and Control Lines	
(10)	**Inked Ribbon**		
	Color:	Black	
	Type:	Cartridge	
	Life Expectancy:	3 million characters	
(11)	**Environmental Conditions**		
	Operating Temperature Range:	41 to 95°F (5 to 35°C)	
	Operating Humidity:	10 to 80% non-condensing	
(12)	**Power Requirement**		
	Voltage:	120 Vac, 60 Hz	
	Current:	1 A maximum	
	Power Consumption:	100 VA maximum	
(13)	**Physical Characteristics**		
	Height:	4.2 inches (107 mm)	
	Width:	14.7 inches (374 mm)	
	Depth:	12.0 inches (305 mm)	
	Weight:	12 pounds (5.5 kg)	

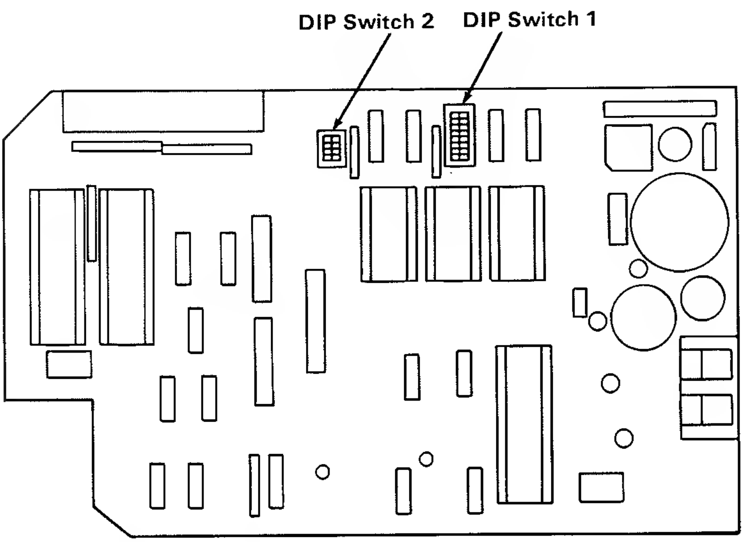
Printer Specifications

(6)	**Printing Characteristics**		
	IBM 80 CPS Matrix Printer		
	Graphics:	64 block characters.	
(6)	**Printing Characteristics**		
	IBM 80 CPS Graphics Printer	Set 1	
	Extra Character Set:	Additional ASCII numbers 160 to 175 contain European characters. Numbers 176 to 223 contain graphic characters. Numbers 224 to 239 contain selected Greek characters. Numbers 240 to 255 contain math and extra symbols.	
		Set 2 The difference in set 2 are ASCII numbers 3, 4, 5, 6, and 21. ASCII numbers 128 to 175 contain European characters.	
	Graphics:	There are 20 block characters and programmable graphics.	
(7)	**Printing Sizes**	Characters per inch	Maximum characters per line
	Subscript:	10	80
	Superscript:	10	80

Additional Printer Specifications

Setting the DIP Switches

There are two DIP switches on the control circuit board. In order to satisfy the user's specific requirements, desired control modes are selectable by the DIP switches. The functions of the switches and their preset conditions at the time of shipment are as shown in the following figures.



Location of Printer DIP Switches

Switch Number	Function	On	Off	Factory-Set Condition
1-1	Not Applicable	—	—	On
1-2	CR	Print Only	Print & Line Feed	On
1-3	Buffer Full	Print Only	Print & Line Feed	Off
1-4	Cancel Code	Invalid	Valid	Off
1-5	Delete Code	Invalid	Valid	On
1-6	Error Buzzer	Sounds	Does Not Sound	On
1-7	Character Generator	N.A.	Graphic Patterns Select	Off
1-8	SLCT IN Signal	Fixed	Not Fixed	On

Functions and Conditions of DIP Switch 1 (Matrix)

Switch Number	Function	On	Off	Factory-Set Condition
2-1	Not Applicable	—	—	On
2-2	Not Applicable	—	—	On
2-3	Auto Feed XT Signal	Fixed Internally	Not Fixed Internally	Off
2-4	Coding Table Select	N.A.	Standard	Off

Functions and Conditions of DIP Switch 2 (Matrix)

Switch Number	Function	On	Off	Factory-Set Condition
1-1	Not Applicable	—	—	On
1-2	CR	Print Only	Print & Line Feed	On
1-3	Buffer Full	Print Only	Print & Line Feed	Off
1-4	Cancel Code	Invalid	Valid	Off
1-5	Not Applicable	—	—	On
1-6	Error Buzzer	Sound	Does Not Sound	On
1-7	Character Generator	Set 2	Set 1	Off
1-8	SLCT IN Signal	Fixed Internally	Not Fixed Internally	On

Functions and Conditions of DIP Switch 1 (Graphics)

Switch Number	Function	On	Off	Factory-Set Condition
2-1	Form Length	12 Inches	11 Inches	Off
2-2	Line Spacing	1/8 Inch	1/6 Inch	Off
2-3	Auto Feed XT Signal	Fixed Internally	Not Fixed Internally	Off
2-4	1 Inch Skip Over Perforation	Valid	Not Valid	Off

Functions and Conditions of DIP Switch 2 (Graphics)

Parallel Interface Description

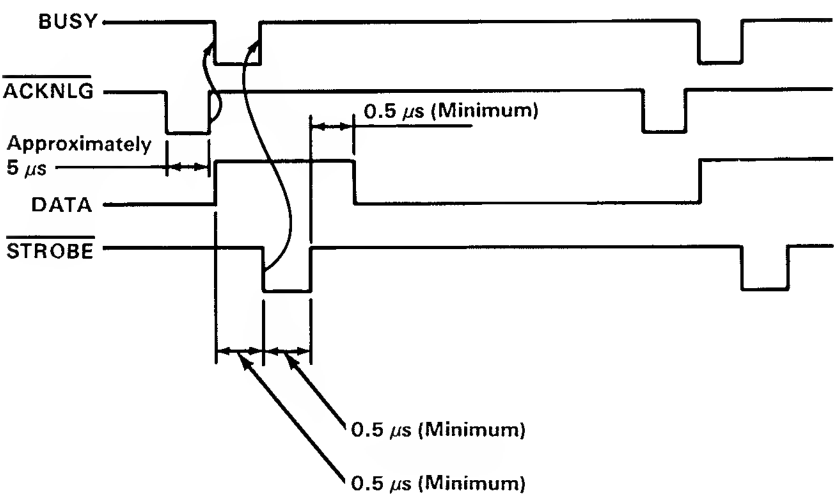
Specifications:

- Data transfer rate: 1000 cps (maximum)
- Synchronization: By externally-supplied $\overline{\text{STROBE}}$ pulses.
- Handshaking $\overline{\text{ACKNLG}}$ or BUSY signals.
- Logic level: Input data and all interface control signals are compatible with the TTL level.

Connector: Plug: 57-30360 (Amphenol)

Connector pin assignment and descriptions of respective interface signals are provided on the following pages.

Data transfer sequence:



Parallel Interface Timing Diagram

Signal Pin No.	Return Pin No.	Signal	Direction	Description
1	19	$\overline{\text{STROBE}}$	In	STROBE pulse to read data in. Pulse width must be more than 0.5 μs at receiving terminal. The signal level is normally "high"; read-in of data is performed at the "low" level of this signal.
2	20	DATA 1	In	These signals represent information of the 1st to 8th bits of parallel data respectively. Each signal is at "high" level when data is logical "1" and "low" when logical "0."
3	21	DATA 2	In	
4	22	DATA 3	In	
5	23	DATA 4	In	
6	24	DATA 5	In	
7	25	DATA 6	In	
8	26	DATA 7	In	
9	27	DATA 8	In	
10	28	$\overline{\text{ACKNLG}}$	Out	Approximately 5 μs pulse; "low" indicates that data has been received and the printer is ready to accept other data.
11	29	BUSY	Out	A "high" signal indicates that the printer cannot receive data. The signal becomes "high" in the following cases: 1. During data entry. 2. During printing operation. 3. In "offline" state. 4. During printer error status.

Connector Pin Assignment and Descriptions of Interface Signals (Part 1 of 3)

Signal Pin No.	Return Pin No.	Signal	Direction	Description
12	30	PE	Out	A "high" signal indicates that the printer is out of paper.
13	—	SLCT	Out	This signal indicates that the printer is in the selected state.
14	—	$\overline{\text{AUTO}}$ $\overline{\text{FEED XT}}$	In	With this signal being at "low" level, the paper is automatically fed one line after printing. (The signal level can be fixed to "low" with DIP SW pin 2-3 provided on the control circuit board.)
15	—	NC		Not used.
16	—	0V		Logic GND level.
17	—	CHASSIS-GND	—	Printer chassis GND. In the printer, the chassis GND and the logic GND are isolated from each other.
18	—	NC	—	Not used.
19-30	—	GND	—	"Twisted-Pair Return" signal; GND level.
31	—	$\overline{\text{INIT}}$	In	When the level of this signal becomes "low" the printer controller is reset to its initial state and the print buffer is cleared. This signal is normally at "high" level, and its pulse width must be more than 50 μs at the receiving terminal.

Connector Pin Assignment and Descriptions of Interface Signals (Part 2 of 3)

Signal Pin No.	Return Pin No.	Signal	Direction	Description
32		$\overline{\text{ERROR}}$	Out	The level of this signal becomes "low" when the printer is in "Paper End" state, "Offline" state and "Error" state.
33	—	GND	—	Same as with pin numbers 19 to 30.
34	—	NC	—	Not used.
35				Pulled up to +5 Vdc through 4.7 k-ohms resistance.
36	—	$\overline{\text{SLCT IN}}$	In	Data entry to the printer is possible only when the level of this signal is "low." (Internal fixing can be carried out with DIP SW 1-8. The condition at the time of shipment is set "low" for this signal.)

Notes: 1. "Direction" refers to the direction of signal flow as viewed from the printer.
2. "Return" denotes "Twisted-Pair Return" and is to be connected at signal-ground level.
When wiring the interface, be sure to use a twisted-pair cable for each signal and never fail to complete connection on the return side. To prevent noise effectively, these cables should be shielded and connected to the chassis of the system unit and printer, respectively.
3. All interface conditions are based on TTL level. Both the rise and fall times of each signal must be less than 0.2 μs.
4. Data transfer must not be carried out by ignoring the $\overline{\text{ACKNLG}}$ or BUSY signal. (Data transfer to this printer can be carried out only after confirming the $\overline{\text{ACKNLG}}$ signal or when the level of the BUSY signal is "low.")

Connector Pin Assignment and Descriptions of Interface Signals (Part 3 of 3)

Printer Modes for the IBM 80 CPS Printers

The IBM 80 CPS Graphics Printer can use any of the combinations listed below, and the print mode can be changed at any place within a line.

The IBM 80 CPS Matrix Printer cannot use the Subscript, Superscript, or Underline print modes. The Double Width print mode will affect the entire line with the matrix printer.

The allowed combinations of print modes that can be selected are listed in the following table. Modes can be selected and combined if they are in the same vertical column.

Printer Modes											
Normal	X	X	X								
Compressed					X	X	X				
Emphasized									X	X	X
Double Strike	X				X				X		
Subscript		X				X				X	
Superscript			X				X				X
Double Width	X	X	X		X	X	X		X	X	X
Underline	X	X	X		X	X	X		X	X	X

Printer Control Codes

On the following pages you will find complete codes for printer characters, controls, and graphics. You may want to keep them handy for future reference. The printer codes are listed in ASCII decimal numeric order (from NUL which is 0 to DEL which is 127). The examples given in the Printer Function descriptions are written in the BASIC language. The "input" description is given when more information is needed for programming considerations.

ASCII decimal values for the printer control codes can be found under "Printer Character Sets."

The descriptions that follow assume that the printer DIP switches have not been changed from their factory settings.

Printer Code	Printer Function
NUL	**Null** Used with ESC B and ESC D as a list terminator. NUL is also used with other printer control codes to select options (for example, ESC S). Example: LPRINT CHR$ (0);
BEL	**Bell** Sounds the printer buzzer for 1 second. Example: LPRINT CHR$ (7);
HT	**Horizontal Tab** Tabs to the next horizontal tap stop. Tab stops are set with ESC D. No tab stops are set when the printer is powered on. (Graphics Printer sets a tab stop every 8 columns when powered on.) Example: LPRINT CHR$ (9);
LF	**Line Feed** Spaces the paper up one line. Line spacing is 1/6-inch unless reset by ESC A, ESC 0, ESC 1, ESC 2 or ESC 3. Example: LPRINT CHR$(10);
VT	**Vertical Tab** Spaces the paper to the next vertical tab position. (Graphics Printer does not allow vertical tabs to be set; therefore, the VT code is treated as LF.) Example: LPRINT CHR$ (11);
FF	**Form Feed** Advances the paper to the top of the next page. **Note:** The location of the paper, when the printer is powered on, determines the top of the page. The next top of page is 11 inches from that position. ESC C can be used to change the page length. Example: LPRINT CHR$ (12);
CR	**Carriage Return** Ends the line that the printer is on and prints the data remaining in the printer buffer. (No Line Feed operation takes place.) **Note:** IBM Personal Computer BASIC adds a Line Feed unless 128 is added [for example, CHR$ (141)]. Example: LPRINT CHR$ (13);

Printer Code	Printer Function
SO	**Shift Out (Double Width)** Changes the printer to the Double Width print mode. **Note:** A Carriage Return, Line Feed or DC4 cancels Double Width print mode. Example: LPRINT CHR$(14);
SI	**Shift In (Compressed)** Changes the printer to the Compressed Character print mode. Example: LPRINT CHR$(15);
DC1	**Device Control 1 (Printer Selected)** (Graphics Printer ignores DC1) Printer accepts data from the system unit. Printer DIP switch 1-8 must be set to the Off position. Example: LPRINT CHR$(17);
DC2	**Device Control 2 (Compressed Off)** Stops printing in the Compressed print mode. Example: LPRINT CHR(18);
DC3	**Device Control 3 (Printer Deselected)** (Graphics Printer ignores DC3) Printer does not accept data from the system unit. The system unit must have the printer select line low, and DIP switch 1-8 must be in the Off position. Example: LPRINT CHR$(19);
DC4	**Device Control 4 (Double Width Off)** Stops printing in the Double Width print mode. Example: LPRINT CHR$(20);
CAN	**Cancel** Clears the printer buffer. Control codes, except SO, remain in effect. Example: LPRINT CHR$ (24);
ESC	**Escape** Lets the printer know that the next data sent is a printer command. (See the following list of commands.) Example: LPRINT CHR$(27);

Printer Code	Printer Function
ESC -	**Escape Minus (Underline)** Format: ESC –;n; (Graphics Printer only) ESC - followed by a 1, prints all of the following data with an underline. ESC - followed by a 0 (zero), cancels the Underline print mode. Example: LPRINT CHR$(27);CHR$(45);CHR$(1);
ESC 0	**Escape Zero (1/8-Inch Line Feeding)** Changes paper feeding to 1/8 inch. Example: LPRINT CHR$(27);CHR$(48);
ESC 1	**Escape 1 (7/72-Inch Line Feeding)** Changes paper feed to 7/72 inch. Example: LPRINT CHR$(27);CHR$(49);
ESC 2	**Escape Two (Starts Variable Line Feeding)** ESC 2 is an execution command for ESC A. If no ESC A command has been given, line feeding returns to 1/6-inch. Example: LPRINT CHR$(27);CHR$(50);
ESC 3	**Escape Three (Variable Line Feeding)** Format: ESC 3;n; (Graphics Printer only) Changes the paper feeding to n/216-inch. The example below sets the paper feeding to 54/216 (1/4) inch. The value of n must be between 1 and 255. Example: LPRINT CHR$(27);CHR$(51);CHR$(54);
ESC 6	**Escape Six (Select Character Set 2)** (Graphics Printer only) Selects character set 2. (See "Printer Character Set 2.") Example: LPRINT CHR$(27);CHR$(54);
ESC 7	**Escape Seven (Select Character Set 1.)** (Graphics Printer only) Selects character set 1. (See "Printer Character Set 1.") Character set 1 is selected when the printer is powered on or reset. Example: LPRINT CHR$(27);CHR$(55);
ESC 8	**Escape Eight (Ignore Paper End)** Allows the printer to print to the end of the paper. The printer ignores the Paper End switch. Example: LPRINT CHR$(27);CHR$(56);

Printer Code	Printer Function
ESC 9	**Escape Nine (Cancel Ignore Paper End)** Cancels the Ignore Paper End command. ESC 9 is selected when the printer is powered on or reset. Example: LPRINT CHR$(27);CHR$(57);
ESC <	**Escape Less Than (Home Head)** (Graphics Printer only) The print head will return to the left margin to print the line following ESC <. This will occur for one line only. Example: LPRINT CHR$(27);CHR$(60);
ESC A	**Escape A (Sets Variable Line Feeding)** Format: ESC A;n; Escape A sets the line-feed to n/72-inch. The example below tells the printer to set line feeding to 24/72-inch. ESC 2 must be sent to the printer before the line feeding will change. For example, ESC A;24 (text) ESC 2 (text). The text following ESC A;24 will space at the previously set line-feed increments. The text following ESC 2 will be printed with new line-feed increments of 24/72-inch. Any increment between 1/72 and 85/72 may be used. Example: LPRINT CHR$(27);CHR$(65);CHR$(24);CHR$(27);CHR$(50);
ESC B	**Escape B (Set Vertical Tabs)** Format: ESC B;n_1;n_2;...n_k;NUL; (Graphics Printer ignores ESC B) Sets vertical tab stop positions. Up to 64 vertical tab stop positions are recognized by the printer. The n's, in the format above, are used to indicate tab stop positions. Tab stop numbers must be received in ascending numeric order. The tab stop numbers will not become valid until the NUL code is entered. Once vertical tab stops are established, they will be valid until new tab stops are specified. (If the printer is reset or powered Off, set tab stops are cleared.) If no tab stop is set, the Vertical Tab command behaves as a Line Feed command. ESC B followed only by NUL will cancel tab stops. The form length must be set by the ESC C command prior to setting tabs. Example: LPRINT CHR$(27);CHR$(66);CHR$(10);CHR$(20);CHR$(40);CHR$(0);

Printer Code	Printer Function
ESC C	**Escape C (Set Lines per Page)** Format: ESC C;n; Sets the page length. The ESC C command must have a value following it to specify the length of page desired. (Maximum form length for the printer is 127 lines.) The example below sets the page length to 55 lines. The printer defaults to 66 lines per page when powered on or reset. Example: LPRINT CHR$(27);CHR$(67);CHR$(55); **Escape C (Set Inches per Page)** Format: ESC C;n;m; (Graphics Printer only) Escape C sets the length of the page in inches. This command requires a value of 0 (zero) for n, and a value between 1 and 22 for m. Example: LPRINT CHR$(27);CHR$(67);CHR$(0);CHR$(12);
ESC D	**Escape D (Set Horizontal Tab Stops)** Format: ESC D;n_1;n_2;...n_k;NUL; Sets the horizontal tab stop positions. The example below shows the horizontal tab stop positions set at printer column positions of 10, 20, and 40. They are followed by CHR$(0), the NUL code. They must also be in ascending numeric order as shown. Tab stops can be set between 1 and 80. When in the Compressed print mode, tab stops can be set up to 132. The maximum number of tabs that can be set is 112. The Graphics Printer can have a maximum of 28 tab stops. The HT (CHR$(9)) is used to execute a tab operation. Example: LPRINT CHR$(27);CHR$(68);CHR$(10)CHR$(20)CHR$(40);CHR$(0);
ESC E	**Escape E (Emphasized)** Changes the printer to the Emphasized print mode. The speed of the printer is reduced to half speed during the Emphasized print mode. Example: LPRINT CHR$(27);CHR$(69);
ESC F	**Escape F (Emphasized Off)** Stops printing in the Emphasized print mode. Example: LPRINT CHR$(27);CHR$(70);
ESC G	**Escape G (Double Strike)** Changes the printer to the Double Strike print mode. The paper is spaced 1/216 of an inch before the second pass of the print head. Example: LPRINT CHR$(27);CHR$(71);

Printer Code	Printer Function
ESC H	**Escape H (Double Strike Off)** Stops printing in the Double Strike mode. Example: LPRINT CHR$(27);CHR$(72);
ESC J	**Escape J (Set Variable Line Feeding)** Format: ESC J;n; (Graphics Printer only) When ESC J is sent to the printer, the paper will feed in increments of n/216 of an inch. The value of n must be between 1 and 255. The example below gives a line feed of 50/216-inch. ESC J is canceled after the line feed takes place. Example: LPRINT CHR$(27);CHR$(74);CHR$(50);
ESC K	**Escape K (480 Bit-Image Graphics Mode)** Format ESC K;n_1;n_2;v_1;v_2;...v_k; (Graphics Printer only) Changes from the Text mode to the Bit-Image Graphics mode. n_1 and n_2 are one byte, which specify the number of bit-image data bytes to be transferred. v_1 through v_k are the bytes of the bit-image data. The number of bit-image data bytes (k) is equal to n_1 +256n_2 and cannot exceed 480 bytes. At every horizontal position, each byte can print up to 8 vertical dots. Bit-image data may be mixed with text data on the same line. **Note:** Assign values to n_1 and n_2 as follows: n_1 represents values from 0 - 255. n_2 represents values from 0 - 1 x 256. MSB is most significant bit and LSB is least significant bit. n_2 MSB LSB 2^{15} 2^{14} 2^{13} 2^{12} 2^{11} 2^{10} 2^{9} 2^{8} n_1 MSB LSB 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0}

Data sent to the printer.

Text (20 characters)	ESC	K	n=360	Bit-image data	Next data

In text mode, 20 characters in text mode correspond to 120 bit-image positions (20 x 6 = 120). The printable portion left in Bit-Image mode is 360 dot positions (480 – 120 = 360).

Data sent to the printer.

		$n_1 n_2$						$n_1 n_2$		
Data A	ESC K	n_1	n_2	Data B	Data C	ESC	K	n_1	n_2	Data D
Text data	Length of data			Bit-image data	Text data	Length of data				Bit-image data

← 480 bit-image dot positions →

Example:

```
TYPE B:GRAPH.TXT
1  'OPEN PRINTER IN RANDOM MODE WITH LENGTH OF 255
2  OPEN "LPT1:" AS #1
3  WIDTH "LPT1:",255
4  PRINT #1,CHR$(13);CHR$(10);
5  SLASH$=CHR$(1)+CHR$(02)+CHR$(04)+CHR$(08)
6  SLASH$=SLASH$+CHR$(16)+CHR$(32)+CHR$(64)+CHR$(128)+CHR$(0)
7  GAP$=CHR$(0)+CHR$(0)+CHR$(0)
8  NDOTS=480
9  'ESC K N1 N2
10 PRINT #1,CHR$(27);"K";CHR$(NDOTS MOD 256);CHR$(FIX (NDOTS/256));
11 ' SEND NDOTS NUMBER OF BIT IMAGE BYTES
12 FOR I=1 TO NDOTS/12 'NUMBER OF SLASHES TO PRINT USING
   GRAPHICS
13 PRINT #1,SLASH$;GAP$;
14 NEXT I
15 CLOSE
16 END
```

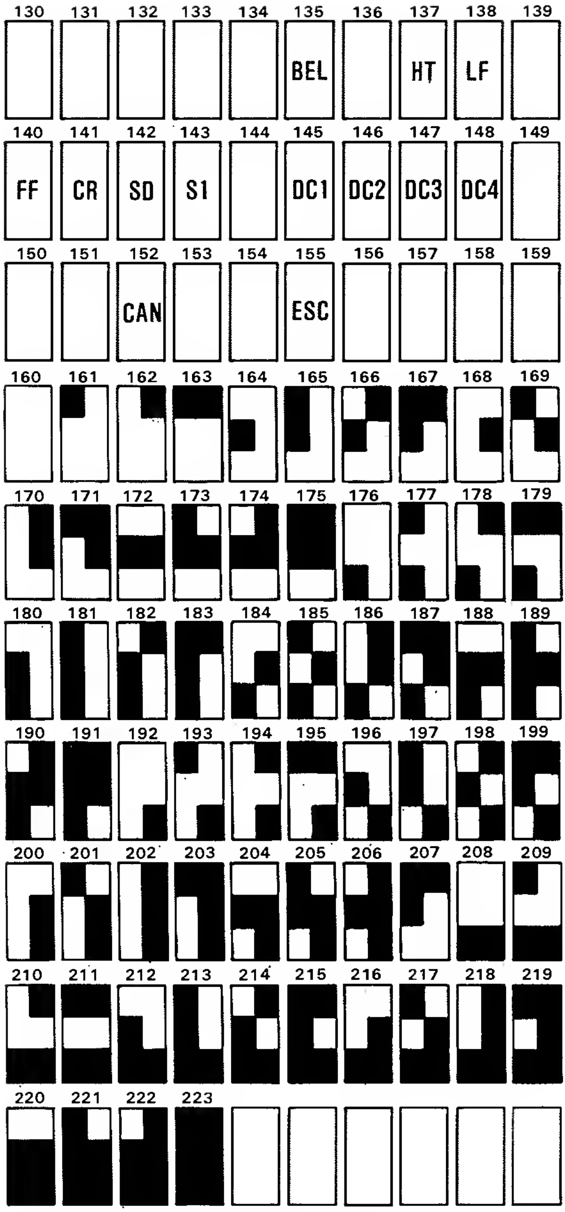
This example will give you a row of slashes printed in the 480 Bit-Image mode.

Printer Code	Printer Function
ESC L	**Escape L (960 Bit-Image Graphics Mode)** Format: ESC L;n_1;n_2;v_1;v_2;...v_k; (Graphics Printer only) Changes from the Text mode to the Bit-Image Graphics mode. The input is similar to ESC K. The 960 Bit-Image mode prints at half the speed of the 480 Bit-Image Graphics mode, but can produce a denser graphic image. The number of bytes of bit-image Data (k) is $n_1 + 256n_2$ but cannot exceed 960. n_1 is in the range of 0 to 255.
ESC N	**Escape N (Set Skip Perforation)** Format ESC N;n; (Graphics Printer only) Sets the Skip Perforation function. The number following ESC N sets the value for the number of lines of Skip Perforation. The example shows a 12-line skip perforation. This will print 54 lines and feed the paper 12 lines. The value of n must be between 1 and 127. ESC N must be reset anytime the page length (ESC C) is changed. Example: CHR$(27);CHR$(78);CHR$(12);
ESC O	**Escape O (Cancel Skip Perforation)** (Graphics Printer only) Cancels the Skip Perforation function. Example: LPRINT CHR$(27);CHR$(79);
ESC S	**Escape S (Subscript/Superscript)** Format: ESC S;n; (Graphics Printer only) Changes the printer to the Subscript print mode when ESC S is followed by a 1, as in the example below. When ESC S is followed by a 0 (zero), the printer will print in the Superscript print mode. Example: LPRINT CHR$(27);CHR$(83);CHR$(1);
ESC T	**Escape T (Subscript/Superscript Off)** (Graphics Printer only) The printer stops printing in the Subscript or Superscript print mode. Example: LPRINT CHR$(27);CHR$(84);
ESC U	**Escape U (Unidirectional Printing)** Format: ESC U;n; (Graphics Printer only) The printer will print from left to right following the input of ESC U;1. When ESC U is followed by a 0 (zero), the left to right printing operation is canceled. The Unidirectional print mode (ESC U) ensures a more accurate print-start position for better print quality. Example: LPRINT CHR$(27);CHR$(85);CHR$(1);

Printer Code	Printer Function
ESC W	**Escape W (Double Width)** Format: ESC W;n; (Graphics Printer only) Changes the printer to the Double Width print mode when ESC W is followed by a 1. This mode is not canceled by a line-feed operation and must be canceled with ESC W followed by a 0 (zero). Example: LPRINT CHR$(27);CHR$(B7);CHR$(1);
ESC Y	**Escape Y (960 Bit-Image Graphics Mode Normal Speed)** Format: ESC Y n_1;n_2;v_1;v_2;...v_k; (Graphics Printer only) Changes from the Text mode to the 960 Bit-Image Graphics mode. The printer prints at normal speed during this operation and cannot print dots on consecutive dot positions. The input of data is similar to ESC L.
ESC Z	**Escape Z (1920 Bit-Image Graphics Mode)** Format: ESC Z;n_1;n_2;v_1;v_2;...v_k; (Graphics Printer only) Changes from the Text mode to the 1920 Bit-Image Graphics mode. The input is similar to the other Bit-Image Graphics modes. ESC Z can print only every third dot position.
DEL	**Delete (Clear Printer Buffer)** (Graphics Printer ignores DEL) Clears the printer buffer. Control codes, except SO, still remain in effect. DIP switch 1-5 must be in the Off position. Example: LPRINT CHR$(127);

0	1	2	3	4	5	6	7	8	9
NUL							BEL		HT
10	11	12	13	14	15	16	17	18	19
LF	VT	FF	CR	SO	SI		DC1	DC2	DC3
20	21	22	23	24	25	26	27	28	29
DC4				CAN			ESC		
30	31	32	33	34	35	36	37	38	39
		SP	!	"	#	$	%	&	'
40	41	42	43	44	45	46	47	48	49
(	)	*	+	,	—	.	/	0	1
50	51	52	53	54	55	56	57	58	59
2	3	4	5	6	7	8	9	:	;
60	61	62	63	64	65	66	67	68	69
<	=	>	?	@	A	B	C	D	E
70	71	72	73	74	75	76	77	78	79
F	G	H	I	J	K	L	M	N	O
80	81	82	83	84	85	86	87	88	89
P	Q	R	S	T	U	V	W	X	Y
90	91	92	93	94	95	96	97	98	99
Z	[	\	]	^	_	`	a	b	c
100	101	102	103	104	105	106	107	108	109
d	e	f	g	h	i	j	k	l	m
110	111	112	113	114	115	116	117	118	119
n	o	p	q	r	s	t	u	v	w
120	121	122	123	124	125	126	127	128	129
x	y	z	{	¦	}	~	DEL	NUL	

Matrix Printer Character Set (Part 1 of 2)



Matrix Printer Character Set (Part 2 of 2)

0	1	2	3	4	5	6	7	8	9
NUL							BEL		HT
10	11	12	13	14	15	16	17	18	19
LF	VT	FF	CR	SO	SI			DC2	
20	21	22	23	24	25	26	27	28	29
DC4				CAN			ESC		
30	31	32	33	34	35	36	37	38	39
		SP	!	"	#	$	%	&	'
40	41	42	43	44	45	46	47	48	49
(	)	*	+	,	—	.	/	0	1
50	51	52	53	54	55	56	57	58	59
2	3	4	5	6	7	8	9	:	;
60	61	62	63	64	65	66	67	68	69
<	=	>	?	@	A	B	C	D	E
70	71	72	73	74	75	76	77	78	79
F	G	H	I	J	K	L	M	N	O
80	81	82	83	84	85	86	87	88	89
P	Q	R	S	T	U	V	W	X	Y
90	91	92	93	94	95	96	97	98	99
Z	[	\	]	^	_	`	a	b	c
100	101	102	103	104	105	106	107	108	109
d	e	f	g	h	i	j	k	l	m
110	111	112	113	114	115	116	117	118	119
n	o	p	q	r	s	t	u	v	w
120	121	122	123	124	125	126	127	128	129
x	y	z	{	¦	}	~		NUL	

Graphics Printer Character Set 1 (Part 1 of 2)

130	131	132	133	134	135	136	137	138	139
					BEL		HT	LF	VT
140	141	142	143	144	145	146	147	148	149
FF	CR	SO	SI			DC2		DC4	
150	151	152	153	154	155	156	157	158	159
		CAN			ESC				
160	161	162	163	164	165	166	167	168	169
á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐
170	171	172	173	174	175	176	177	178	179
¬	½	¼	¡	«	»	░	▒	▓	│
180	181	182	183	184	185	186	187	188	189
┤	╡	╢	╖	╕	╣	║	╗	╝	╜
190	191	192	193	194	195	196	197	198	199
╛	┐	└	┴	┬	├	─	┼	╞	╟
200	201	202	203	204	205	206	207	208	209
╚	╔	╩	╦	╠	═	╬	╧	╨	╤
210	211	212	213	214	215	216	217	218	219
╥	╙	╘	╒	╓	╫	╪	┘	┌	█
220	221	222	223	224	225	226	227	228	229
▄	▌	▐	▀	α	β	Γ	Π	Σ	σ
230	231	232	233	234	235	236	237	238	239
μ	τ	Φ	Θ	Ω	δ	∞	Ø	ϵ	$\cap$
240	241	242	243	244	245	246	247	248	249
$\equiv$	$\pm$	$\geq$	$\leq$	⌠	⌡	$\div$	$\approx$	°	▪
250	251	252	253	254	255				
–	$\sqrt{}$	^{n}	^{2}	■	SP				

Graphics Printer Character Set 1 (Part 2 of 2)

0	1	2	3	4	5	6	7	8	9
NUL			♥	♦	♣	♠	BEL		HT
10	11	12	13	14	15	16	17	18	19
LF	VT	FF	CR	SO	SI			DC2	
20	21	22	23	24	25	26	27	28	29
DC4	§			CAN			ESC		
30	31	32	33	34	35	36	37	38	39
		SP	!	"	#	$	%	&	'
40	41	42	43	44	45	46	47	48	49
(	)	*	+	,	—	.	/	0	1
50	51	52	53	54	55	56	57	58	59
2	3	4	5	6	7	8	9	:	;
60	61	62	63	64	65	66	67	68	69
<	=	>	?	@	A	B	C	D	E
70	71	72	73	74	75	76	77	78	79
F	G	H	I	J	K	L	M	N	O
80	81	82	83	84	85	86	87	88	89
P	Q	R	S	T	U	V	W	X	Y
90	91	92	93	94	95	96	97	98	99
Z	[	\	]	^	_	`	a	b	c
100	101	102	103	104	105	106	107	108	109
d	e	f	g	h	i	j	k	l	m
110	111	112	113	114	115	116	117	118	119
n	o	p	q	r	s	t	u	v	w
120	121	122	123	124	125	126	127	128	129
x	y	z	{	¦	}	~		ç	ü

Graphics Printer Character Set 2 (Part 1 of 2)

130	131	132	133	134	135	136	137	138	139
é	â	ä	à	å	ç	ê	ë	è	ï
140	141	142	143	144	145	146	147	148	149
î	ì	Ä	Â	É	æ	Æ	ô	ö	ò
150	151	152	153	154	155	156	157	158	159
û	ù	ÿ	ö	ü	¢	£	¥	₧	ƒ
160	161	162	163	164	165	166	167	168	169
á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐
170	171	172	173	174	175	176	177	178	179
¬	½	¼	¡	«	»	░	▒	▓	│
180	181	182	183	184	185	186	187	188	189
┤	╡	╢	╖	╕	╣	║	╗	╝	╜
190	191	192	193	194	195	196	197	198	199
╛	┐	└	┴	┬	├	─	┼	╞	╟
200	201	202	203	204	205	206	207	208	209
╚	╔	╩	╦	╠	═	╬	╧	╨	╤
210	211	212	213	214	215	216	217	218	219
╥	╙	╘	╒	╓	╫	╪	┘	┌	█
220	221	222	223	224	225	226	227	228	229
▄	▌	▐	▀	α	β	Γ	Π	Σ	σ
230	231	232	233	234	235	236	237	238	239
μ	τ	Φ	Θ	Ω	δ	∞	ø	∈	∩
240	241	242	243	244	245	246	247	248	249
≡	±	≥	≤	⌠	⌡	÷	≈	°	▪
250	251	252	253	254	255				
-	√	^{n}	^{2}	■	SP				

Graphics Printer Character Set 2 (Part 2 of 2)

IBM Printer Adapter

The printer adapter is specifically designed to attach printers with a parallel port interface, but it can be used as a general input/output port for any device or application that matches its input/output capabilities. It has 12 TTL-buffer output points, which are latched and can be written and read under program control using the processor In or Out instruction. The adapter also has five steady-state input points that may be read using the processor's In instructions.

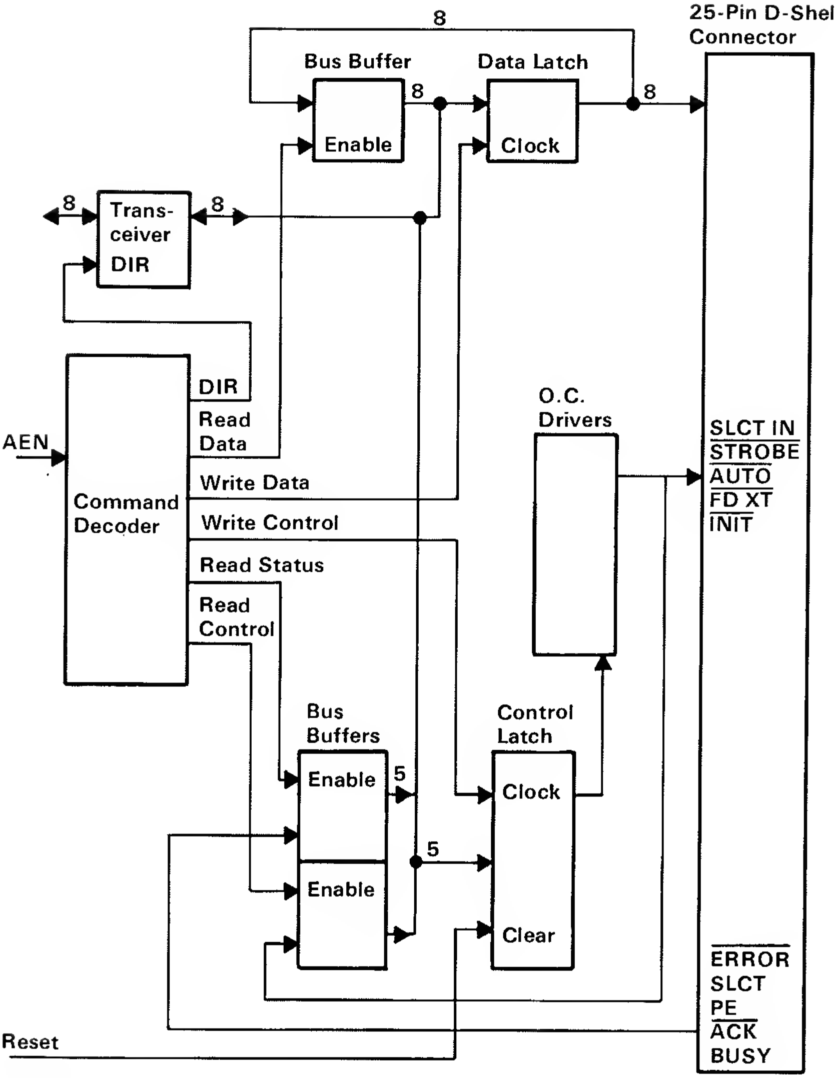
In addition, one input can also be used to create a processor interrupt. This interrupt can be enabled and disabled under program control. Reset from the power-on circuit is also ORed with a program output point, allowing a device to receive a power-on reset when the processor is reset.

The input/output signals are made available at the back of the adapter through a right-angled, PCB-mounted, 25-pin, D-shell connector. This connector protrudes through the rear panel of the system or expansion unit, where a cable may be attached.

When this adapter is used to attach a printer, data or printer commands are loaded into an 8-bit, latched, output port, and the strobe line is activated, writing data to the printer. The program then may read the input ports for printer status indicating when the next character can be written, or it may use the interrupt line to indicate "not busy" to the software.

The output ports may also be read at the card's interface for diagnostic loop functions. This allows faults to be isolated between the adapter and the attaching device.

This same function is also part of the combination IBM Monochrome Display and Printer Adapter. A block diagram of the printer adapter is on the next page.



Printer Adapter Block Diagram

Programming Considerations

The printer adapter responds to five I/O instructions: two output and three input. The output instructions transfer data into 2 latches whose outputs are presented on pins of a 25-pin D-shell connector.

Two of the three input instructions allow the processor to read back the contents of the two latches. The third allows the processor to read the real time status of a group of pins on the connector.

A description of each instruction follows.

IBM Monochrome Display & Printer Adapter				**Printer Adapter**			
Output to address hex 3BC				Output to address hex 378			
Bit 7 Pin 9	Bit 6 Pin 8	Bit 5 Pin 7	Bit 4 Pin 6	Bit 3 Pin 5	Bit 2 Pin 4	Bit 1 Pin 3	Bit 0 Pin 2

The instruction captures data from the data bus and is present on the respective pins. These pins are each capable of sourcing 2.6 mA and sinking 24 mA.

It is essential that the external device not try to pull these lines to ground.

IBM Monochrome Display & Printer Adapter		**Printer Adapter**			
Output to address hex 3BE		Output to address hex 37A			
	Bit 4 IRQ Enable	$\overline{\text{Bit 3}}$ Pin 17	Bit 2 Pin 16	$\overline{\text{Bit 1}}$ Pin 14	$\overline{\text{Bit 0}}$ Pin 1

This instruction causes the latch to capture the five least significant bits of the data bus. The four least significant bits present their outputs, or inverted versions of their outputs, to the respective pins shown above. If bit 4 is written as 1, the card will interrupt the processor on the condition that pin 10 transitions high to low.

These pins are driven by open collector drivers pulled to +5 Vdc through 4.7 k-ohm resistors. They can each sink approximately 7 mA and maintain 0.8 volts down-level.

IBM Monochrome Display & Printer Adapter	Printer Adapter
Input from address hex 3BC	Input from address hex 378

This command presents the processor with data present on the pins associated with the out to hex 3BC. This should normally reflect the exact value that was last written to hex 3BC. If an external device should be driving data on these pins (in violation of usage ground rules) at the time of an input, this data will be ORed with the latch contents.

IBM Monochrome Display & Printer Adapter	Printer Adapter
Input from address hex 3BD	Input from address hex 379

This command presents realtime status to the processor from the pins as follows.

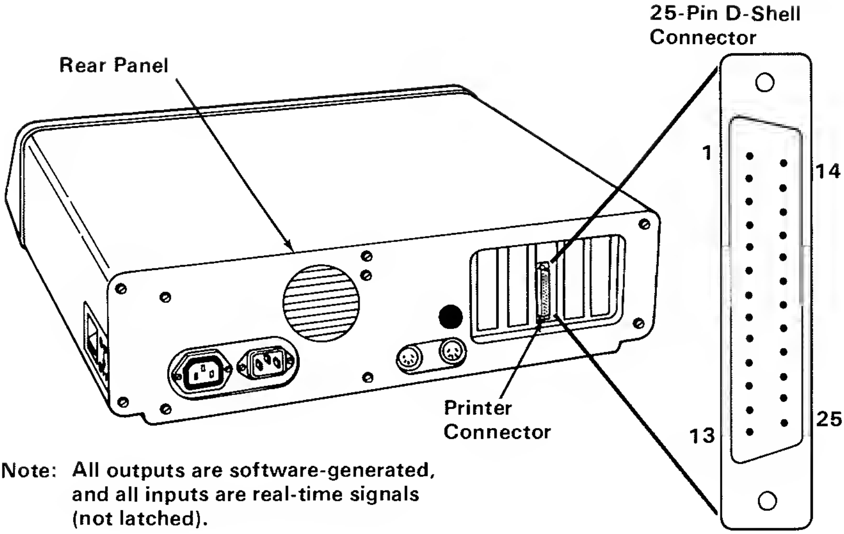
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pin 11	Pin 10	Pin 12	Pin 13	Pin 15	—	—	—

IBM Monochrome Display & Printer Adapter	**Printer Adapter**
Input from address hex 3BE	Input from address hex 37A

This instruction causes the data present on pins 1, 14, 16, 17, and the IRQ bit to read by the processor. In the absence of external drive applied to these pins, data read by the processor will exactly match data last written to hex 3BE in the same bit positions. Note that data bits 0-2 are not included. If external drivers are dotted to these pins, that data will be ORed with data applied to the pins by the hex 3BE latch.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			IRQ Enable	$\overline{\text{Pin 17}}$	Pin 16	$\overline{\text{Pin 14}}$	$\overline{\text{Pin 1}}$
			Por=0	Por=1	Por=0	Por=1	Por=1

These pins assume the states shown after a reset from the processor.



Note: All outputs are software-generated, and all inputs are real-time signals (not latched).

At Standard TTL Levels

Printer ↔ Printer Adapter

Signal Name	Adapter Pin Number
-Strobe	1
+Data Bit 0	2
+Data Bit 1	3
+Data Bit 2	4
+Data Bit 3	5
+Data Bit 4	6
+Data Bit 5	7
+Data Bit 6	8
+Data Bit 7	9
-Acknowledge	10
+Busy	11
+P.End (out of paper)	12
+Select	13
-Auto Feed	14
-Error	15
-Initialize Printer	16
-Select Input	17
Ground	18-25

Connector Specifications

IBM Monochrome Display and Printer Adapter

This chapter has two functions. The first is to provide the interface to the IBM Monochrome Display. The second provides a parallel interface for the IBM 80 CPS Printer. This second function is fully discussed in the "IBM Printer Adapter" section.

The monitor adapter is designed around the Motorola 6845 CRT controller module. There are 4K bytes of static memory on the adapter which is used for the display buffer. This buffer has two ports and may be accessed directly by the processor. No parity is provided on the display buffer.

Two bytes are fetched from the display buffer in 553 ns, providing a data rate of 1.8M bytes/second.

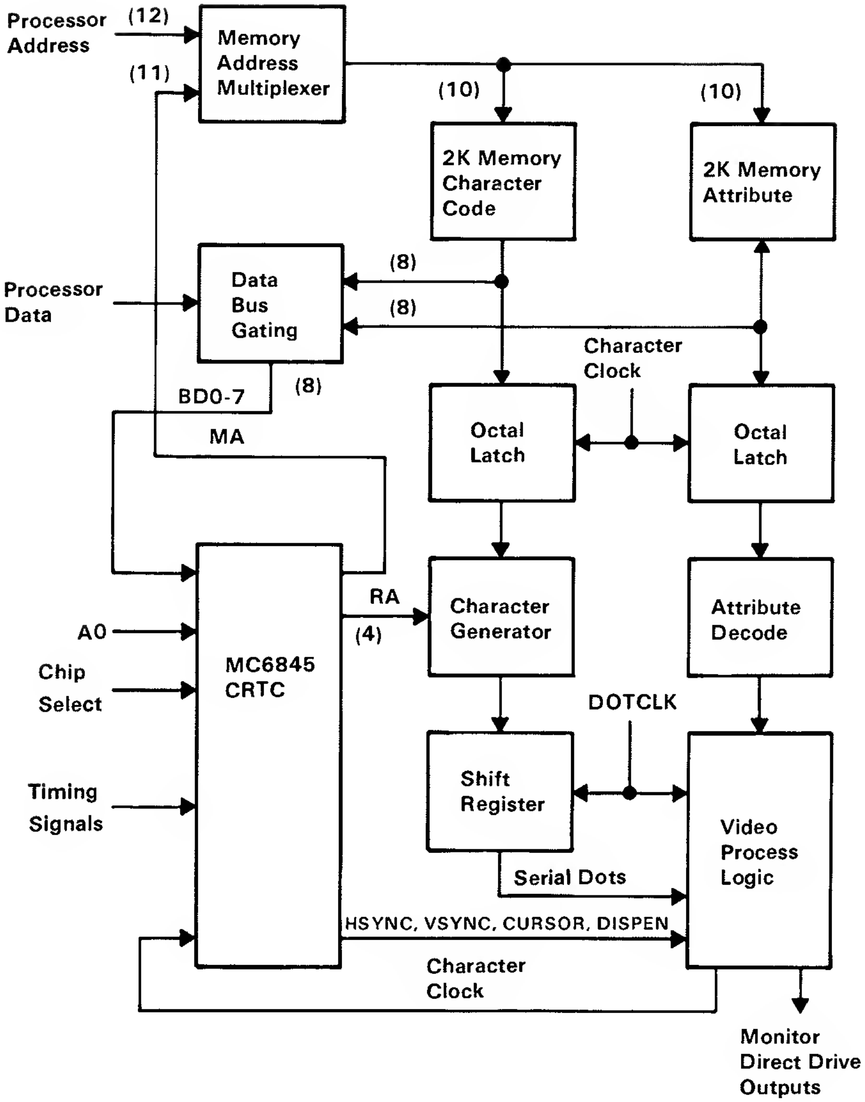
The monitor adapter supports 256 different character codes. An 8K-byte character generator contains the fonts for the character codes. The characters, values, and screen characteristics are given in "Appendix C: Of Characters, Keystrokes, and Color."

This monitor adapter, when used with a display containing P39 phosphor, will not support a light pen.

Where possible, only one low-power Schottky (LS) load is present on any I/O slot. Some of the address bus lines have two LS loads. No signal has more than two LS loads.

Characteristics of the monitor adapter are listed below:

- 80 by 25 screen
- Direct-drive output
- 9 by 14 character box
- 7 by 9 character
- 18 kHz monitor
- Character attributes



IBM Monochrome Display Adapter Block Diagram

Programming Considerations

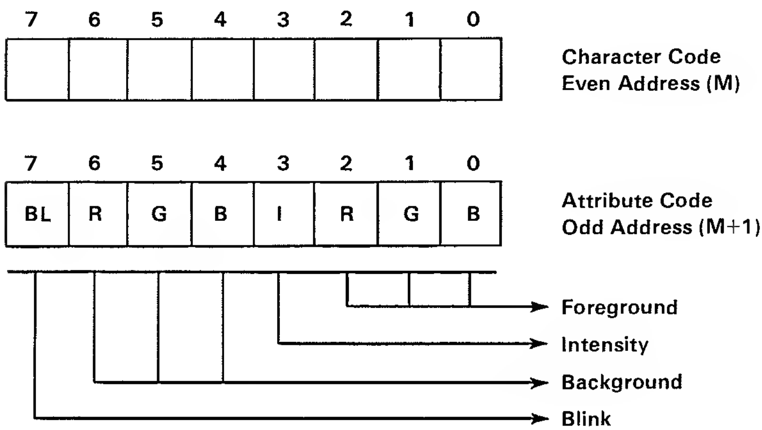
The following table summarizes the 6845 internal data registers, their functions, and their parameters. For the IBM Monochrome Display, the values must be programmed into the 6845 to ensure proper initialization of the device.

Register Number	Register File	Program Unit	IBM Monochrome Display (Address in hex)
R0	Horizontal Total	Characters	61
R1	Horizontal Displayed	Characters	50
R2	Horizontal Sync Position	Characters	52
R3	Horizontal Sync Width	Characters	F
R4	Vertical Total	Character Rows	19
R5	Vertical Total Adjust	Scan Line	6
R6	Vertical Displayed	Character Row	19
R7	Vertical Sync Position	Character Row	19
R8	Interlace Mode	--------	02
R9	Maximum Scan Line Address	Scan Line	D
R10	Cursor Start	Scan Line	B
R11	Cursor End	Scan Line	C
R12	Start Address (H)	--------	00
R13	Start Address (L)	--------	00
R14	Cursor (H)	--------	00
R15	Cursor (L)	--------	00
R16	Reserved	--------	--
R17	Reserved	--------	--

To ensure proper initialization, the first command issued to the attachment must be to send to CRT control port 1 (hex 3B8), a hex 01, to set the high-resolution mode. If this bit is not set, then the processor access to the monochrome adapter must never occur. If the high-resolution bit is not set, the processor will stop running.

System configurations that have both an IBM Monochrome Display Adapter and Printer Adapter, and an IBM Color/Graphics Monitor Adapter, must ensure that both adapters are properly initialized after a power-on reset. Damage to either display may occur if not properly initialized.

The IBM Monochrome Display and Printer Adapter supports 256 different character codes. In the character set are alphanumerics and block graphics. Each character in the display buffer has a corresponding character attribute. The character code must be an even address, and the attribute code must be an odd address in the display buffer.



The adapter decodes the character attribute byte as defined above. The blink and intensity bits may be combined with the foreground and background bits to further enhance the character attribute functions listed below.

Background R G B	Foreground R G B	Function
0 0 0	0 0 0	Non-Display
0 0 0	0 0 1	Underline
0 0 0	1 1 1	White Character/Black Background
1 1 1	0 0 0	Reverse Video

The 4K display buffer supports one screen of 25 rows of 80 characters, plus a character attribute for each display character. The starting address of the buffer is hex B0000. The display buffer can be read from using DMA; however, at least one wait-state will be inserted by the processor. The duration of the wait-state will vary, because the processor/monitor access is synchronized with the character clock on this adapter.

Interrupt level 7 is used on the parallel interface. Interrupts can be enabled or disabled through the printer control port. The interrupt is a high-level active signal.

The figure below breaks down the functions of the I/O address decode for the adapter. The I/O address decode is from hex 3B0 through hex 3BF. The bit assignment for each I/O address follows:

I/O Register Address	Function
3B0	Not Used
3B1	Not Used
3B2	Not Used
3B3	Not Used
3B4*	6845 Index Register
3B5*	6845 Data Register
3B6	Not Used
3B7	Not Used
3B8	CRT Control Port 1
3B9	Reserved
3BA	CRT Status Port
3BB	Reserved
3BC	Parallel Data Port
3BD	Printer Status Port
3BE	Printer Control Port
3BF	Not Used

*The 6845 Index and Data Registers are used to program the CRT controller to interface the high-resolution IBM Monochrome Display.

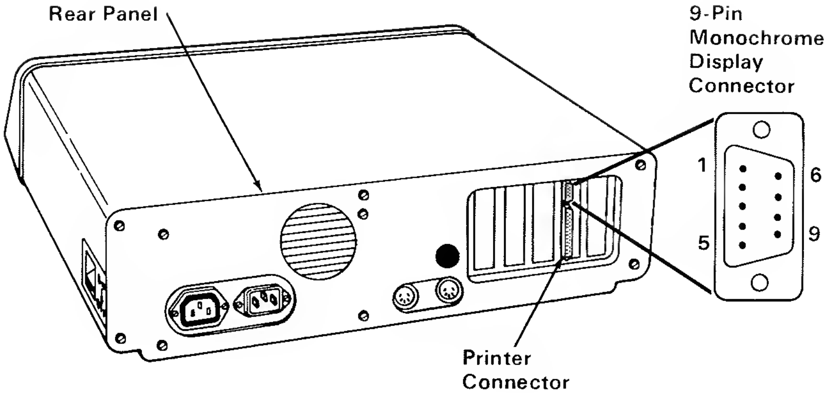
I/O Address and Bit Map

Bit Number	Function
0	+High Resolution Mode
1	Not Used
2	Not Used
3	+Video Enable
4	Not Used
5	+Enable Blink
6,7	Not Used

6845 CRT Control Port 1 (Hex 3B8)

Bit Number	Function
0	+Horizontal Drive
1	Reserved
2	Reserved
3	+Black/White Video

6845 CRT Status Port (Hex 3BA)



At Standard TTL Levels

IBM Monochrome Display	Signal	Pin	IBM Monochrome Display and Printer Adapter
	Ground	1	
	Ground	2	
	Not Used	3	
	Not Used	4	
	Not Used	5	
←	+Intensity	6	
←	+Video	7	
←	+Horizontal	8	
←	-Vertical	9	

Note: Signal voltages are 0.0 to 0.6 Vdc at down level and +2.4 to 3.5 Vdc at high level.

Connector Specifications

Notes:

IBM Monochrome Display

The high-resolution IBM Monochrome Display attaches to the system unit through two cables approximately 3 feet (914 millimeters) in length. One cable is a signal cable that contains the direct drive interface from the IBM Monochrome Display and Printer Adapter.

The second cable provides ac power to the display from the system unit. This allows the system-unit power switch to also control the display unit. An additional benefit is a reduction in the requirements for wall outlets to power the system. The display contains an 11-½ inch (283 millimeters), diagonal 90° deflection CRT. The CRT and analog circuits are packaged in an enclosure so the display may either sit on top of the system unit or on a nearby tabletop or desk. The unit has both brightness and contrast adjustment controls on the front surface that are easily accessible to the operator.

Operating Characteristics

Screen

- High-persistence green phosphor (P 39).
- Etched surface to reduce glare.
- Size is 80 characters by 25 lines.
- Character box is 9 dots wide by 14 dots high.

Video Signal

- Maximum bandwidth of 16.257 MHz.

Vertical Drive

- Screen refreshed at 50 Hz with 350 lines of vertical resolution and 720 lines of horizontal resolution.

Horizontal Drive

- Positive-level, TTL-compatibility at a frequency of 18.432 kHz.

IBM Color/Graphics Monitor Adapter

The IBM Color/Graphics Monitor Adapter is designed to attach to the IBM Color Display, to a variety of television-frequency monitors, or to home television sets (user-supplied RF modulator is required for home television sets). The adapter is capable of operating in black-and-white or color. It provides three video interfaces: a composite-video port, a direct-drive port, and a connection interface for driving a user-supplied RF modulator. In addition, a light pen interface is provided.

The adapter has two basic modes of operation: alphanumeric (A/N) and all-points-addressable graphics (APA). Additional modes are available within the A/N and APA modes. In the A/N mode, the display can be operated in either a 40-column by 25-row mode for a low-resolution monitor or home television, or in an 80-column by 25-row mode for high-resolution monitors. In both modes, characters are defined in an 8-wide by 8-high character box and are 7-wide by 7-high, with one line of descender for lowercase characters. Both uppercase and lowercase characters are supported in all modes.

The character attributes of reverse video, blinking, and highlighting are available in the black-and-white mode. In the color mode, sixteen foreground and eight background colors are available for each character. In addition, blinking on a per-character basis is available.

The monitor adapter contains 16K bytes of storage. As an example, a 40-column by 25-row display screen uses 1000 bytes to store character information, and 1000 bytes to store attribute/color information. This would mean that up to eight display screens can be stored in the adapter memory. Similarly, in an 80-column by 25-row mode, four display screens may be stored in the adapter. The entire 16K bytes of storage on the display adapter are directly addressable by the processor, which allows maximum software flexibility in managing the screen.

In A/N color modes, it is also possible to select the color of the screen's border. One of sixteen colors can be selected.

In the APA mode, there are two resolutions available: a medium-resolution color graphics mode (320 PELs by 200 rows) and a high-resolution black-and-white graphics mode (640 PELs by 200 rows). In the medium-resolution mode, each picture element (PEL) may have one of four colors. The background color (color 0) may be any of the 16 possible colors. The remaining three colors come from one of the two software-selectable palettes. One palette contains green/red/brown; the other contains cyan/magenta/white.

The high-resolution mode is available only in black-and-white because the entire 16K bytes of storage in the adapter is used to define the on or off of the PELs.

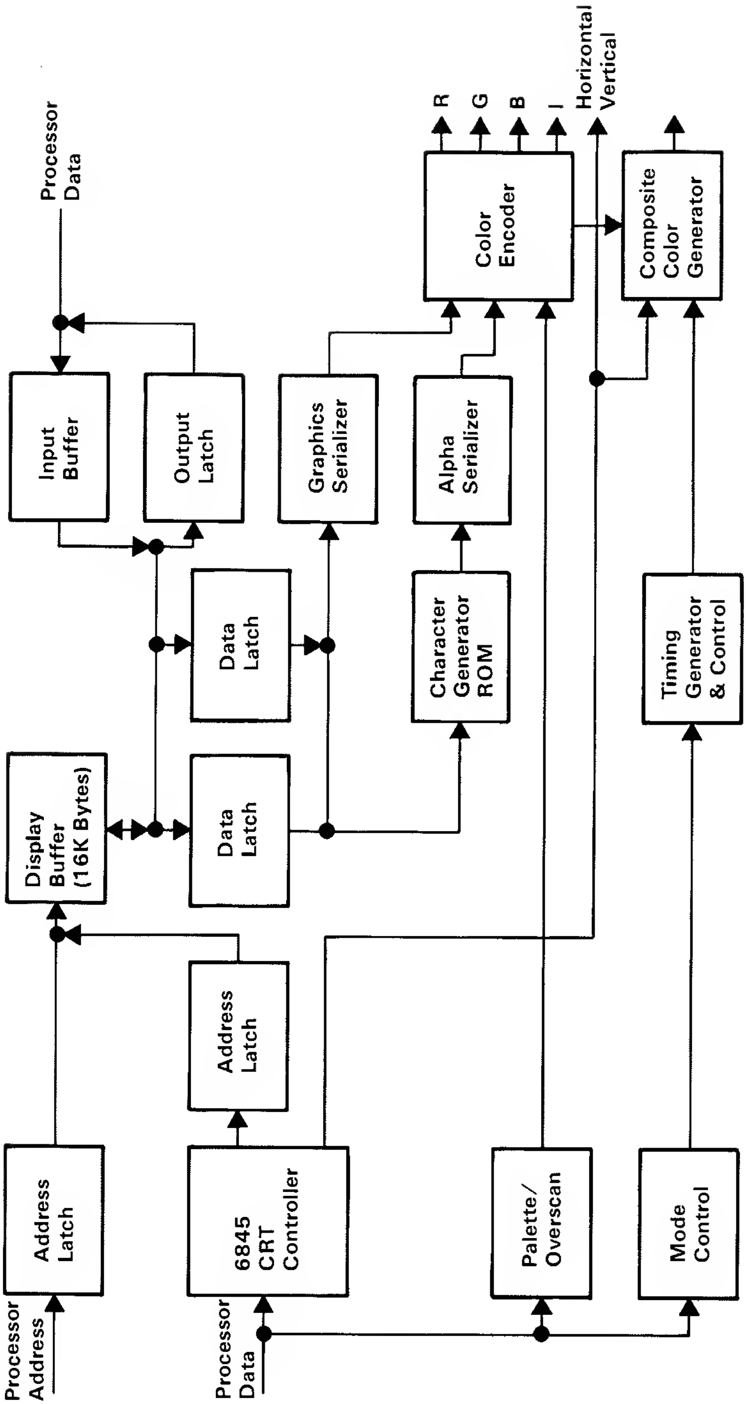
The adapter operates in noninterlace mode at either 7 or 14 MHz, depending on the mode of operation selected.

In the A/N mode, characters are formed from a ROM character generator. The character generator contains dot patterns for 256 different characters. The character set contains the following major groupings of characters:

- 16 special characters for game support
- 15 characters for word-processing editing support
- 96 characters for the standard ASCII graphics set
- 48 characters for foreign-language support
- 48 characters for business block-graphics support (allowing drawing of charts, boxes, and tables using single and double lines)
- 16 selected Greek characters
- 15 selected scientific-notation characters

The color/graphics monitor adapter function is packaged on a single card. The direct-drive and composite-video ports are right-angle mounted connectors on the adapter, and extend through the rear panel of the unit. The direct-drive video port is a 9-pin D-shell female connector. The composite-video port is a standard female phono-jack.

The display adapter is implemented using a Motorola 6845 CRT controller device. This adapter is highly programmable with respect to raster and character parameters. Therefore, many additional modes are possible with clever programming of the adapter.



Color/Graphics Monitor Adapter Block Diagram

Descriptions of Major Components

Motorola 6845 CRT Controller

This device provides the necessary interface to drive a raster-scan CRT.

Mode Set Register

This is a general-purpose, programmable, I/O register. It has I/O ports that may be individually programmed. Its function in this attachment is to provide mode selection and color selection in the medium-resolution color-graphics mode.

Display Buffer

The display buffer resides in the processor-address space, starting at address hex B8000. It provides 16K bytes of dynamic read/write memory. A dual-ported implementation allows the processor and the graphics control unit to access the buffer. The processor and the CRT control unit have equal access to this buffer during all modes of operation, except in the high-resolution alphanumeric mode. In this mode, only the processor should access to this buffer during the horizontal-retrace intervals. While the processor may write to the required buffer at any time, a small amount of display interference will result if this does not occur during the horizontal-retrace intervals.

Character Generator

This attachment utilizes a ROM character generator. It consists of 8K bytes of storage that cannot be read from or written to under software control. This is a general-purpose ROM character generator with three different character fonts. Two character fonts are used on the color/graphics adapter: a 7-high by 7-wide double-dot font and a 5-wide by 7-high single-dot font. The font is selected by a jumper (P3). The single-dot font is selected by inserting the jumper; the double-dot font is selected by removing the jumper.

Timing Generator

This generator produces the timing signals used by the 6845 CRT controller and by the dynamic memory. It also resolves the processor/graphic controller contentions for accessing the display buffer.

Composite Color Generator

This generator produces base band video color information.

Alphanumeric Mode

Every display-character position in the alphanumeric mode is defined by two bytes in the regen buffer (a part of the monitor adapter), not the system memory. Both the color/graphics and the monochrome display adapter use the following 2-byte character/attribute format.

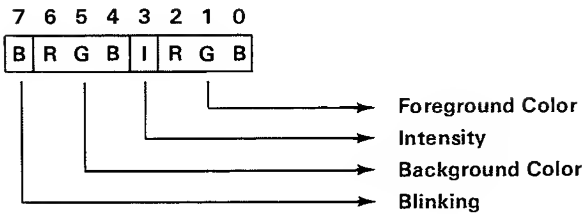
Display-Character Code Byte	Attribute Byte
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

The functions of the attribute byte are defined by the following table:

Attribute Function	Attribute Byte							
	7	6	5	4	3	2	1	0
	B	R	G	B	I	R	G	B
	FG	Background			Foreground			
Normal	B	0	0	0	I	1	1	1
Reverse Video	B	1	1	1	I	0	0	0
Nondisplay (Black)	B	0	0	0	I	0	0	0
Nondisplay (White)	B	1	1	1	I	1	1	1

I = Highlighted Foreground (Character)
B = Blinking Foreground (Character)

The attribute byte definitions are:



In the alphanumeric mode, the display mode can be operated in either a low-resolution mode or a high-resolution mode.

The low-resolution alphanumeric mode has the following features:

- Supports home color televisions or low-resolution monitors
- Displays up to 25 rows of 40 characters each
- ROM character generator that contains dot patterns for a maximum of 256 different characters
- Requires 2,000 bytes of read/write memory (on the adapter)
- Character box is 8-high by 8-wide
- Two jumper-controlled character fonts are available:
  5-wide by 7-high single-dot character font with one descender
  7-wide by 7-high double-dot character font with one descender
- One character attribute for each character

The high-resolution alphanumeric mode has the following features:

- Supports the IBM Color Display or other color monitor with direct-drive input capability
- Supports a black-and-white composite-video monitor
- Displays up to 25 rows of 80 characters each

- ROM displays generator that contains dot patterns for a maximum of 256 different characters
- Requires 4,000 bytes of read/write memory (on the adapter)
- Character box is 8-high by 8-wide
- Two jumper-controlled character fonts are available:
  5-wide by 7-high single-dot character font with one descender
  7-wide by 7-high double-dot character font with one descender
- One character attribute for each character

Monochrome vs Color/Graphics Character Attributes

Foreground and background colors are defined by the attribute byte of each character, whether using the IBM Monochrome Display and Printer Adapter or the IBM Color/Graphics Monitor Adapter. The following table describes the colors for each adapter:

Attribute Byte 7	6	5	4	3	2	1	0	Monochrome Display Adapter		Color/Graphics Monitor Adapter	
B	R	G	B	I	R	G	B	Background Color	Character Color	Background Color	Character Color
FG	Background			Foreground							
B	0	0	0	I	1	1	1	Black	White	Black	White
B	1	1	1	I	0	0	0	White	Black	White	Black
B	0	0	0	I	0	0	0	Black	Black	Black	Black
B	1	1	1	I	1	1	1	White	White	White	White

The monochrome display adapter will produce white characters on a white background with any other code. The color/graphics adapter will change foreground and background colors according to the color value selected. The color values for the various red, green, blue, and intensity bit settings are given in the following table.

R	G	B	I	Color
0	0	0	0	Black
0	0	1	0	Blue
0	1	0	0	Green
0	1	1	0	Cyan
1	0	0	0	Red
1	0	1	0	Magenta
1	1	0	0	Brown
1	1	1	0	White
0	0	0	1	Gray
0	0	1	1	Light Blue
0	1	0	1	Light Green
0	1	1	1	Light Cyan
1	0	0	1	Light Red
1	0	1	1	Light Magenta
1	1	0	1	Yellow
1	1	1	1	White (High Intensity)

Code written with an underline attribute for the IBM Monochrome Display, when executed on a color/graphics monitor adapter, will result in a blue character where the underline attribute is encountered. Also, code written on a color/graphics monitor adapter with blue characters will be displayed as white characters on a black background, with a white underline on the IBM Monochrome Display.

Remember that not all monitors recognize the intensity (I) bit.

Graphics Mode

The IBM Color/Graphics Monitor Adapter has three modes available within the graphics mode. They are low-resolution color graphics, medium-resolution color graphics, and high-resolution color graphics. However, only medium- and high-resolution graphics are supported in ROM. The following table summarizes the three modes.

Mode	Horizontal (PELs)	Vertical (Rows)	Number of Colors Available (Includes Background Color)
Low Resolution	160	100	16 (Includes black-and-white)
Medium Resolution	320	200	4 Colors Total 1 of 16 for Background and 1 of Green, Red, or Brown or 1 of Cyan, Magenta, or White
High Resolution	640	200	Black-and-white only

Low-Resolution Color-Graphics Mode

The low-resolution mode supports home television or color monitors. This mode is not supported in ROM. It has the following features:

- Contains a maximum of 100 rows of 160 PELs, with each PEL being 2-high by 2-wide
- Specifies 1 of 16 colors for each PEL by the I, R, G, and B bits
- Requires 16,000 bytes of read/write memory (on the adapter)
- Uses memory-mapped graphics

Medium-Resolution Color-Graphics Mode

The medium-resolution mode supports home televisions or color monitors. It has the following features:

- Contains a maximum of 200 rows of 320 PELs, with each PEL being 1-high by 1-wide
- Preselects one of four colors for each PEL
- Requires 16,000 bytes of read/write memory (on the adapter)
- Uses memory-mapped graphics

- Formats 4 PELs per byte in the following table:

7	6	5	4	3	2	1	0
C1	C0	C1	C0	C1	C0	C1	C0
First Display PEL		Second Display PEL		Third Display PEL		Fourth Display PEL	

- Organizes graphics storage in two banks of 8,000 bytes, using the following format:

Memory Address (in hex)	Function
B8000	Even Scans (0,2,4,...198) 8,000 bytes
B9F3F	Not Used
BA000	Odd Scans (1,3,5...199) 8,000 Bytes
BBF3F	Not Used
BBFFF	

Address hex B8000 contains PEL instruction for the upper-left corner of the display area.

- Color selection is determined by the following logic:

C1	C0	Function
0	0	Dot takes on the color of 1 of 16 preselected background colors
0	1	Selects first color of preselected Color Set 1 or Color Set 2
1	0	Selects second color of preselected Color Set 1 or Color Set 2
1	1	Selects third color of preselected Color Set 1 or Color Set 2

C1 and C0 will select 4 of 16 preselected colors. This color selection (palette) is preloaded in an I/O port.

Tow two colors sets are:

Color Set 1	Color Set 2
Color 1 is Green Color 2 is Red Color 3 is Brown	Color 1 is Cyan Color 2 is Magenta Color 3 is White

The background colors are the same basic 8 colors as defined for low-resolution graphics, plus 8 alternate intensities defined by the intensity bit, for a total of 16 colors, including black and white.

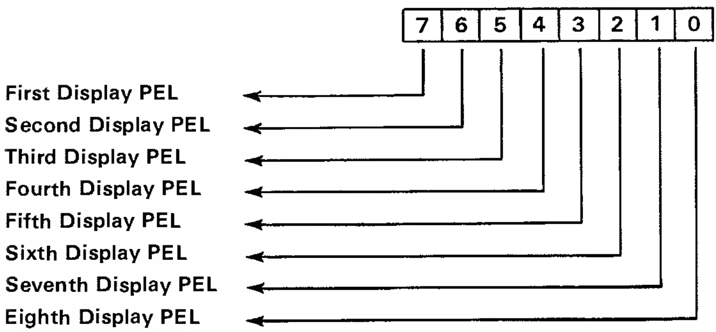
High-Resolution Black-and-White Graphics Mode

The high-resolution mode supports color monitors. This mode has the following features:

- Contains a maximum of 200 rows of 640 PELs, with each PEL being 1-high by 1-wide.
- Supports black-and-white mode only.
- Requires 16,000 bytes of read/write memory (on the adapter).

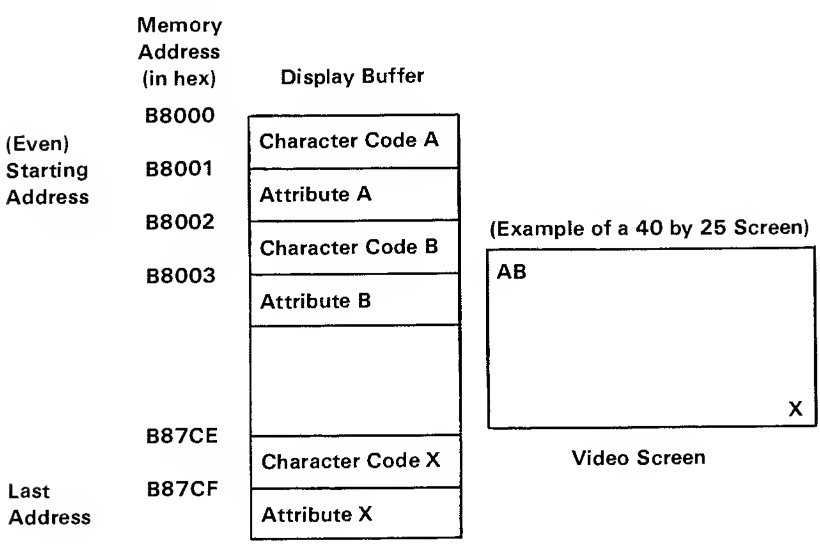
- Addressing and mapping procedures are the same as medium-resolution color graphics, but the data format is different. In this mode, each bit in memory is mapped to a PEL on the screen.

- Formats 8 PELs per byte in the following manner:



Description of Basic Operations

In the alphanumeric mode, the adapter fetches character and attribute information from its display buffer. The starting address of the display buffer is programmable through the 6845, but it must be an even address. The character codes and attributes are then displayed according to their relative positions in the buffer.



The processor and the display control unit have equal access to the display buffer during all the operating modes, except the high-resolution alphanumeric mode. During this mode, the processor should access the display buffer during the vertical retrace time. If it does not, the display will be affected with random patterns as the processor is using the display buffer. In the alphanumeric mode, the characters are displayed from a prestored ROM character generator that contains the dot patterns of all the displayable characters.

In the graphics mode, the displayed dots and colors (up to 16K bytes) are also fetched from the display buffer. The bit configuration for each graphics mode is explained in "Graphics Mode."

I	R	G	B	Color
0	0	0	0	Black
0	0	0	1	Blue
0	0	1	0	Green
0	0	1	1	Cyan
0	1	0	0	Red
0	1	0	1	Magenta
0	1	1	0	Brown
0	1	1	1	White
1	0	0	0	Gray
1	0	0	1	Light Blue
1	0	1	0	Light Green
1	0	1	1	Light Cyan
1	1	0	0	Light Red
1	1	0	1	Light Magenta
1	1	1	0	Yellow
1	1	1	1	High Intensity White

Note: "I" provides extra luminance (brightness) to each available shade. This results in the light colors listed above, except for monitors that do not recognize the "I" bit.

Summary of Available Colors

Programming Considerations

Programming the 6845 CRT Controller

The 6845 has 19 accessible internal registers, which are used to define and control a raster-scan CRT display. One of these registers, the Index register, is actually used as a pointer to the other 18 registers. It is a write-only register, which is loaded from the processor by executing an 'out' instruction to I/O address hex 3D4. The five least significant bits of the I/O bus are loaded into the Index register.

In order to load any of the other 18 registers, the Index register is first loaded with the necessary pointer; then the Data Register is loaded with the information to be placed in the selected register. The Data Register is loaded from the processor by executing an Out instruction to I/O address hex 3D5.

The following table defines the values that must be loaded into the 6845 CRT Controller registers to control the different modes of operation supported by the attachment:

Address Register	Register Number	Register Type	Units	I/O	40 by 25 Alpha-numeric	80 by 25 Alpha-numeric	Graphic Modes
0	R0	Horizontal Total	Character	Write Only	38	71	38
1	R1	Horizontal Displayed	Character	Write Only	28	50	28
2	R2	Horizontal Sync Position	Character	Write Only	2D	5A	2D
3	R3	Horizontal Sync Width	Character	Write Only	0A	0A	0A
4	R4	Vertical Total	Character Row	Write Only	1F	1F	7F
5	R5	Vertical Total Adjust	Scan Line	Write Only	06	06	06
6	R6	Vertical Displayed	Character Row	Write Only	19	19	64
7	R7	Vertical Sync Position	Character Row	Write Only	1C	1C	70
8	R8	Interlace Mode	-	Write Only	02	02	02
9	R9	Maximum Scan Line Address	Scan Line	Write Only	07	07	01
A	R10	Cursor Start	Scan Line	Write Only	06	06	06
B	R11	Cursor End	Scan Line	Write Only	07	07	07
C	R12	Start Address (H)	-	Write Only	00	00	00
D	R13	Start Address (L)	-	Write Only	00	00	00
E	R14	Cursor Address (H)	-	Read/ Write	XX	XX	XX
F	R15	Cursor Address (L)	-	Read/ Write	XX	XX	XX
10	R16	Light Pen (H)	-	Read Only	XX	XX	XX
11	R17	Light Pen (L)	-	Read Only	XX	XX	XX

Note: All register values are given in hexadecimal

6845 Register Description

Programming the Mode Control and Status Register

The following I/O devices are defined on the color/graphics adapter.

Hex Address	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Function of Register
3D8	1	1	1	1	0	1	1	0	0	0	Mode Control Register (D0)
3D9	1	1	1	1	0	1	1	0	0	1	Color Select Register (D0)
3DA	1	1	1	1	0	1	1	0	1	0	Status Register (D1)
3DB	1	1	1	1	0	1	1	0	1	1	Clear Light Pen Latch
3DC	1	1	1	1	0	1	1	1	0	0	Preset Light Pen Latch
3D4	1	1	1	1	0	1	0	Z	Z	0	6845 Index Register
3D5	1	1	1	1	0	1	0	Z	Z	1	6845 Data Register
3D0	1	1	1	1	0	1	0	Z	Z	0	6845 Registers
3D1	1	1	1	1	0	1	0	Z	Z	1	6845 Registers

Z = don't care condition

Color-Select Register

This is a 6-bit output-only register (cannot be read). Its I/O address is hex 3D9, and it can be written to by using the 8088 I/O Out command.

Bit 0	Selects B (Blue) Border Color in 40 x 25 Alphanumeric Mode Selects B (Blue) Background Color in 320 x 200 Graphics Mode Selects B (Blue) Foreground Color in 640 x 200 Graphics Mode
Bit 1	Selects G (Green) Border Color in 40 x 25 Alphanumeric Mode Selects G (Green) Background Color in 320 x 200 Graphics Mode Selects G (Green) Foreground Color in 640 x 200 Graphics Mode
Bit 2	Selects R (Red) Border Color in 40 x 25 Alphanumeric Mode Selects R (Red) Background Color in 320 x 200 Graphics Mode Selects R (Red) Foreground Color in 640 x 200 Graphics Mode
Bit 3	Selects I (Intensified) Border Color in 40 x 25 Alphanumeric Mode Selects I (Intensified) Background Color in 320 x 200 Graphics Mode Selects I (Intensified) Foreground Color in 640 x 200 Graphics Mode
Bit 4	Selects Alternate, Intensified Set of Colors in Graphics Mode Selects Background Colors in the Alphanumeric Mode
Bit 5	Selects Active Color Set in 320 x 200 Graphics Mode
Bit 6	Not Used
Bit 7	Not Used

Bits 0, 1, 2, 3 These bits select the screen's border color in the 40 by 25 alphanumeric mode. They select the screen's background color (C0-C1) in the medium-resolution (320 by 200) color-graphics mode.

Bits 4 This bit, when set, will select an alternate, intensified set of colors. Selects background colors in the alphanumeric mode.

Bit 5 This bit is only used in the medium-resolution (320 by 200) color-graphics mode. It is used to select the active set of screen colors for the display.

When bit 5 is set to 1, colors are determined as follows:

C1	C0	Set Selected
0	0	Background (Defined by bits 0-3 of port hex 3D9)
0	1	Cyan
1	0	Magenta
1	1	White

When bit 5 is set to 0, colors are determined as follows:

C1	C0	Set Selected
0	0	Background (Defined by bits 0-3 of port hex 3D9)
0	1	Green
1	0	Red
1	1	Brown

Mode-Select Register

This is a 6-bit output-only register (cannot be read). Its I/O address is hex 3D8, and it can be written to using the 8088 I/O Out command.

The following is a description of the register's functions:

Bit 0	80 x 25 Alphanumeric Mode
Bit 1	Graphics Select
Bit 2	Black/White Select
Bit 3	Enable Video Signal
Bit 4	High-Resolution (640 x 200) Black/White Mode
Bit 5	Change Background Intensity to Blink Bit
Bit 6	Not Used
Bit 7	Not Used

Bit 0 A 1 selects 80 by 25 alphanumeric mode
A 0 selects 40 by 25 alphanumeric mode

Bit 1 A 1 selects 320 by 200 graphics mode
A 0 selects alphanumeric mode

Bit 2 A 1 selects black-and-white mode
A 0 selects color mode

Bit 3 A 1 enables the video signal at certain times when modes are being changed. The video signal should be disabled when changing modes.

Bit 4 A 1 selects the high-resolution (640 by 200) black-and-white graphics mode. One color of 8 can be selected on direct-drive sets in this mode by using register hex 3D9.

Bit 5 When on, this bit will change the character background intensity to the blinking attribute function for alphanumeric modes. When the high-order attribute bit is not selected, 16 background colors (or intensified colors) are available. For normal operation, this bit should be set to 1 to allow the blinking function.

Mode Register Summary

Bits

0	1	2	3	4	5	
0	0	1	1	0	1	40 x 25 Alphanumeric Black-and-White
0	0	0	1	0	1	40 x 25 Alphanumeric Color
1	0	1	1	0	1	80 x 25 Alphanumeric Black-and-White
1	0	0	1	0	1	80 x 25 Alphanumeric Color
0	1	1	1	0	z	320 x 200 Black-and-White Graphics
0	1	0	1	0	z	320 x 200 Color Graphics
0	1	1	1	1	z	640 x 200 Black-and-White Graphics

Enable Blink Attribute
640 x 200 Black-and-White
Enable Video Signal
Select Black-and-White Mode
Select 320 x 200 Graphics
80 x 25 Alphanumeric Select

z = don't care condition

Note: The low-resolution (160 by 100) mode requires special programming and is set up as the 40 by 25 alphanumeric mode.

Status Register

The status register is a 4-bit read-only register. Its I/O address is hex 3DA, and it can be read using the 8088 I/O In instruction. The following is a description of the register functions:

Bit 0	Display Enable
Bit 1	Light-Pen Trigger Set
Bit 2	Light-Pen Switch Made
Bit 3	Vertical Sync
Bit 4	Not Used
Bit 5	Not Used
Bit 6	Not Used
Bit 7	Not Used

Bit 0 This bit, when active, indicates that a regen buffer memory access can be made without interfering with the display.

Bit 1 This bit, when active, indicates that a positive-going edge from the light-pen has set the light pen's trigger. This trigger is reset upon power-on and may also be cleared by performing an I/O Out command to hex address 3DB. No specific data setting is required; the action is address-activated.

Bit 2 The light-pen switch status is reflected in this status bit. The switch is not latched or debounced. A 0 indicates that the switch is on.

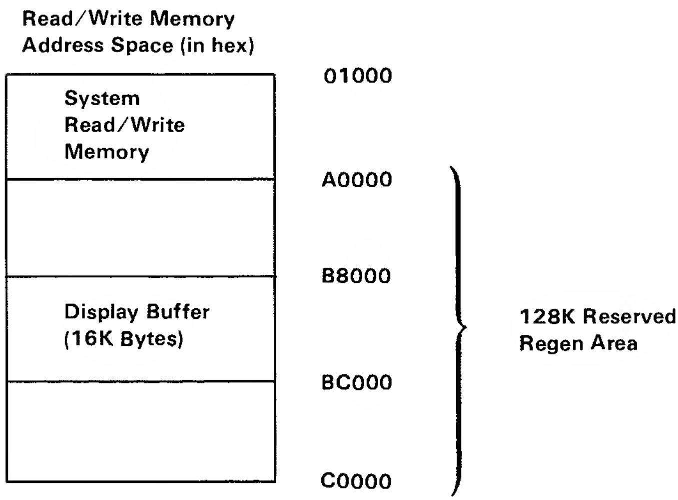
Bit 3 This bit, when active, indicates that the raster is in a vertical retrace mode. This is a good time to perform screen-buffer updating.

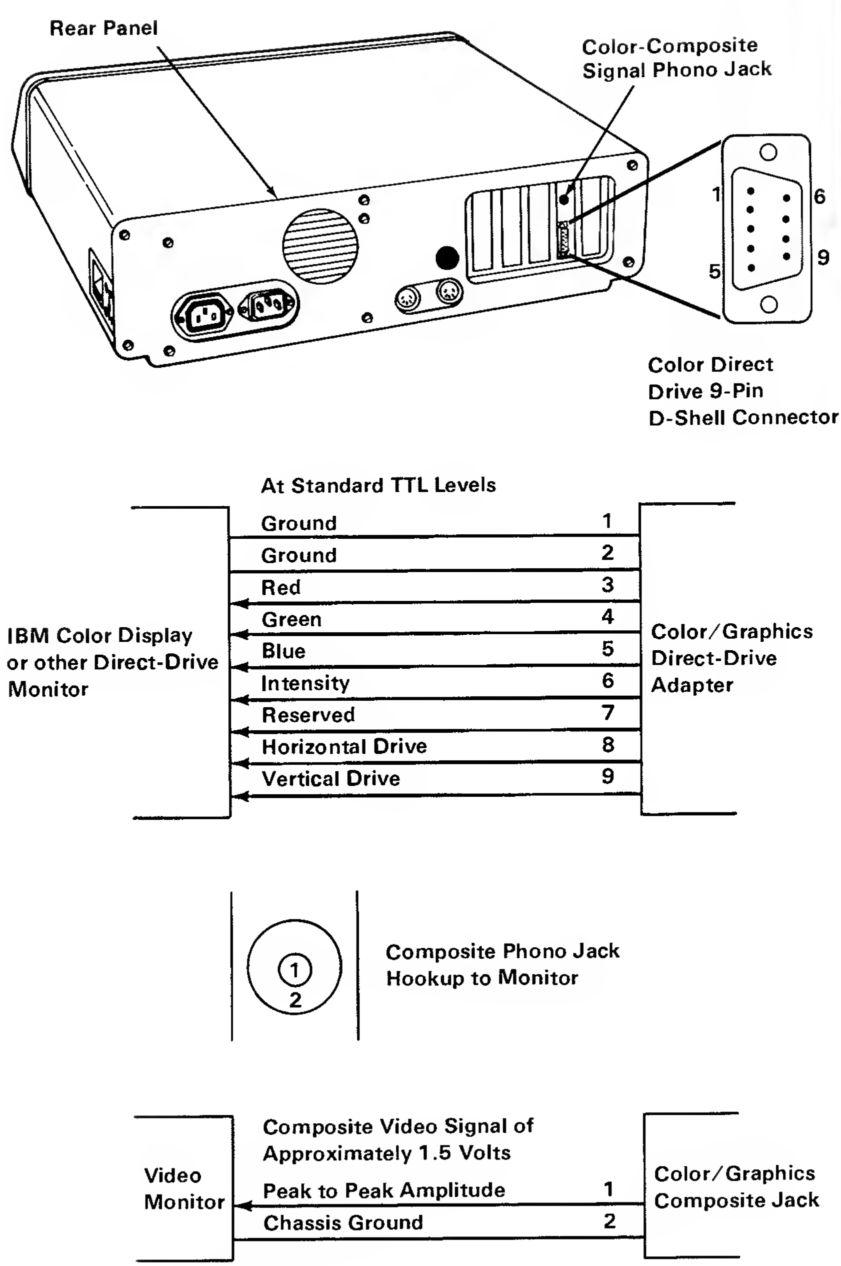
Sequence of Events for Changing Modes

1. Determine the mode of operation.
2. Reset 'video enable' bit in mode-select register.
3. Program 6845 to select mode.
4. Program mode/color select registers including re-enabling video.

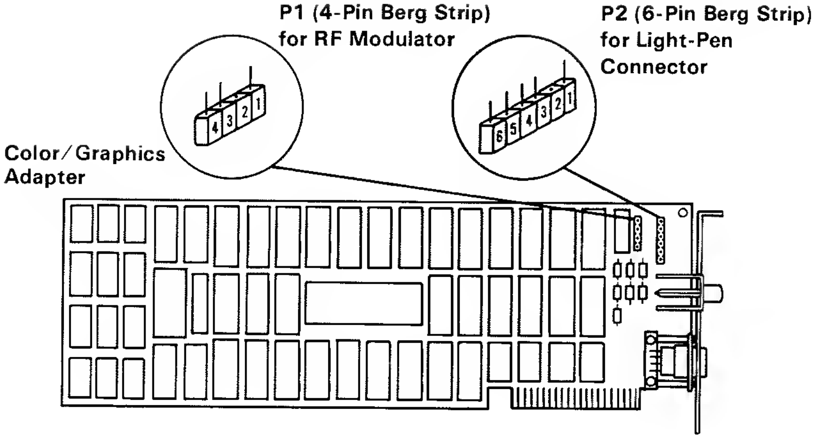
Memory Requirements

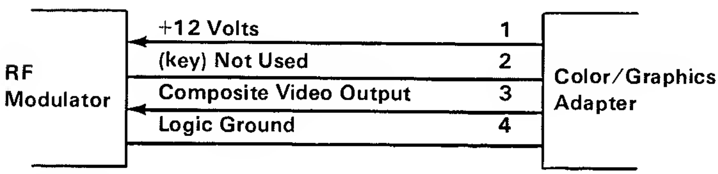
The memory used by this adapter is self-contained. It consists of 16K bytes of memory without parity. This memory is used as both a display buffer for alphanumeric data and as a bit map for graphics data. The regen buffer's address starts at hex B8000.



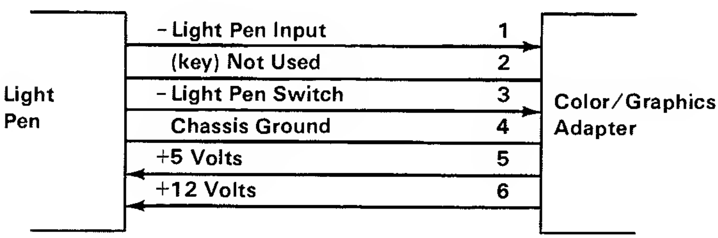


Connector Specifications (Part 1 of 2)





RF Modulator Interface



Light Pen Interface

Connector Specifications (Part 2 of 2)

IBM Color Display

The IBM Color Display attaches to the system unit by a signal cable that is approximately 5 feet (1.5 meters) in length. This signal cable provides a direct-drive interface from the IBM Color/Graphics Monitor Adapter.

A second cable provides ac power to the display from a standard wall outlet. The display has its own power control and indicator. The display will accept either 120-volt 60-Hz, or 220-volt 50-Hz power. The power supply in the display automatically switches to match the applied power.

The display has a 13-inch (340 millimeters) CRT. The CRT and analog circuits are packaged in an enclosure so the display may sit either on top of the system unit or on a nearby tabletop or desk. Front panel controls and indicators include: Power-On control, Power-On indicator, Brightness and Contrast controls. Two additional rear-panel controls are the Vertical Hold and Vertical Size controls.

Operating Characteristics

Screen

- High contrast (black) screen.
- Displays up to 16 colors, when used with the IBM Color/Graphics Monitor Adapter.
- Characters defined in an 8-high by 8-wide matrix.

Video Signal

- Maximum video bandwidth of 14 MHz.
- Red, green, and blue video signals and intensity are all independent.

Vertical Drive

- Screen refreshed at 60 Hz with 200 vertical lines of resolution.

Horizontal Drive

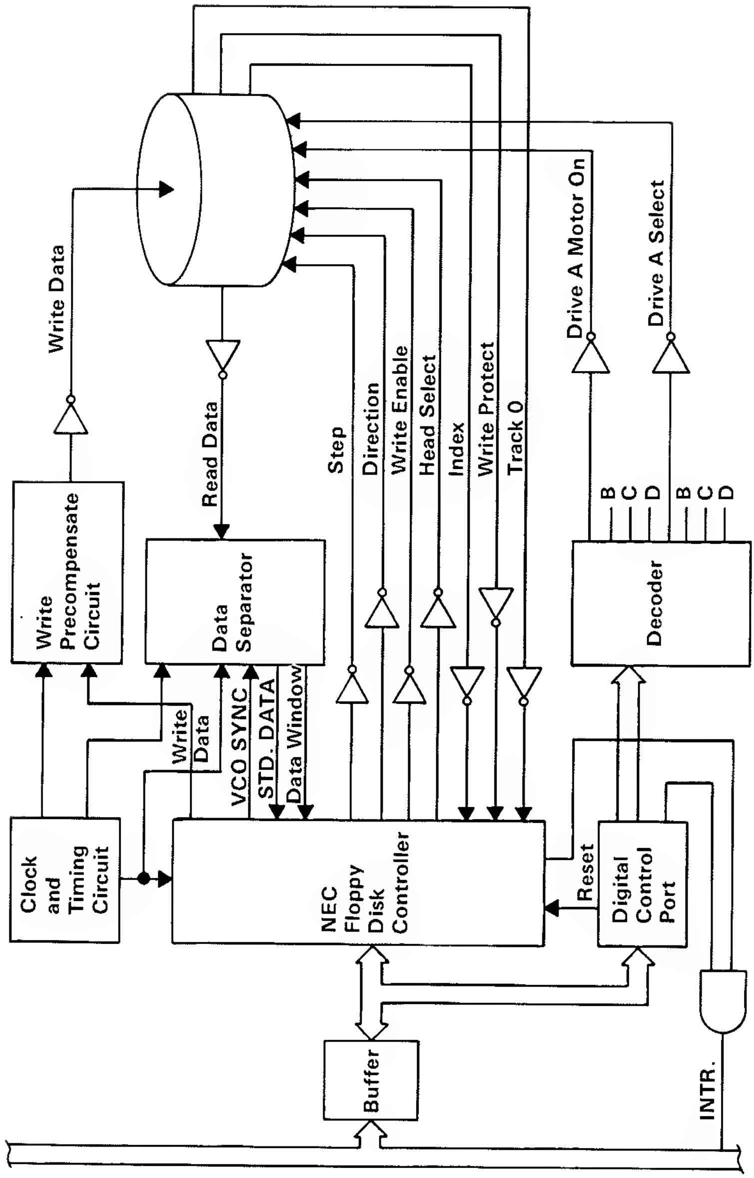
- Positive-level, TTL-compatibility, at a frequency of 15.75 kHz.

IBM 5-1/4″ Diskette Drive Adapter

The 5-1/4 inch diskette drive adapter fits into one of the expansion slots in the system unit. It attaches to one or two diskette drives through an internal, daisy-chained flat cable that connects to one end of the drive adapter. The adapter has a connector at the other end that extends through the rear panel of the system unit. This connector has signals for two additional external diskette drives; thus the 5-1/4 inch diskette drive adapter can attach four 5-1/4 inch drives – two internal and two external.

The adapter is designed for double-density, MFM-coded, diskette drives and uses write precompensation with an analog phase-lock loop for clock and data recovery. The adapter is a general-purpose device using the NEC μPD765 compatible controller. Therefore, the diskette drive parameters are programmable. In addition, the attachment supports the diskette drive's write-protect feature. The adapter is buffered on the I/O bus and uses the system board's direct memory access (DMA) for record data transfers. An interrupt level is also used to indicate when an operation is complete and that a status condition requires processor attention.

In general, the 5-1/4 inch diskette drive adapter presents a high-level command interface to software I/O drivers. A block diagram of the 5-1/4 inch diskette drive adapter is on the following page.



5-1/4 Inch Diskette Drive Adapter Block Diagram

Functional Description

From a programming point of view, this attachment consists of an 8-bit digital-output register in parallel with an NEC μPD765 or equivalent floppy disk controller (FDC).

In the following description, drive numbers 0, 1, 2, and 3 are equivalent to drives A, B, C, and D.

Digital-Output Register

The digital-output register (DOR) is an output-only register used to control drive motors, drive selection, and feature enable. All bits are cleared by the I/O interface reset line. The bits have the following functions:

Bits 0 and 1 These bits are decoded by the hardware to select one drive if its motor is on:

Bit 1	Bit 0	Drive
0	0	0 (A)
0	1	1 (B)
1	0	2 (C)
1	1	3 (D)

Bit 2 The FDC is held reset when this bit is clear. It must be set by the program to enable the FDC.

Bit 3 This bit allows the FDC interrupt and DMA requests to be gated onto the I/O interface. If this bit is cleared, the interrupt and DMA request I/O interface drivers are disabled.

Bits 4, 5, 6, and 7 These bits control, respectively, the motors of drives 0, 1, 2 (A, B, C), and 3 (D). If a bit is clear, the associated motor is off, and the drive cannot be selected.

Floppy Disk Controller

The floppy disk controller (FDC) contains two registers that may be accessed by the main system processor: a status register and a data register. The 8-bit main status register contains the status information of the FDC and may be accessed at any time. The 8-bit data register (actually consisting of several registers in a stack with only one register presented to the data bus at a time) stores data, commands, parameters, and provides floppy disk drive (FDD) status information. Data bytes are read from or written to the data register in order to program or obtain results after a particular command. The main status register may only be read and is used to facilitate the transfer of data between the processor and FDC.

The bits in the main status register (hex 34F) are defined as follows:

Bit Number	Name	Symbol	Description
DB0	FDD A Busy	DAB	FDD number 0 is in the Seek mode.
DB1	FDD B Busy	DBB	FDD number 1 is in the Seek mode.
DB2	FDD C Busy	DCB	FDD number 2 is in the Seek mode.
DB3	FDD D Busy	DDB	FDD number 3 is in the Seek mode.
DB4	FDC Busy	CB	A read or write command is in process.
DB5	Non-DMA Mode	NDM	The FDC is in the non-DMA mode.
DB6	Data Input/ Output	DIO	Indicates direction of data transfer between FDC and processor. If DIO = "1," then transfer is from FDC data register to the processor. If DIO = "0," then transfer is from the processor to FDC data register.
DB7	Request for Master	RQM	Indicates data register is ready to send or receive data to or from the processor. Both bits DIO and RQM should be used to perform the handshaking functions of "ready" and "direction" to the processor.

The FDC is capable of performing 15 different commands. Each command is initiated by a multi-byte transfer from the processor, and the result after execution of the command may also be a multi-byte transfer back to the processor. Because of this multi-byte interchange of information between the FDC and the processor, it is convenient to consider each command as consisting of three phases:

Command Phase

The FDC receives all information required to perform a particular operation from the processor.

Execution Phase

The FDC performs the operation it was instructed to do.

Result Phase

After completion of the operation, status and other housekeeping information is made available to the processor.

Programming Considerations

The following tables define the symbols used in the command summary, which follows.

Symbol	Name	Description
A0	Address Line 0	A0 controls selection of main status register (A0 = 0) or data register (A0 = 1).
C	Cylinder Number	C stands for the current/selected cylinder (track) number of the medium.
D	Data	D stands for the data pattern that is going to be written into a sector.
D7-D0	Data Bus	8-bit data bus, where D7 stands for a most significant bit, and D0 stands for a least significant bit.
DTL	Data Length	When N is defined as 00, DTL stands for the data length that users are going to read from or write to the sector.
EOT	End of Track	EOT stands for the final sector number on a cylinder.
GPL	Gap Length	GPL stands for the length of gap 3 (spacing between sectors excluding VCO sync field).
H	Head Address	H stands for head number 0 or 1, as specified in ID field.
HD	Head	HD stands for a selected head number 0 or 1. (H = HD in all command words.)
HLT	Head Load Time	HLT stands for the head load time in the FDD (4 to 512 ms in 4-ms increments).
HUT	Head Unload Time	HUT stands for the head unload time after a read or write operation has occurred (0 to 480 ms in 32-ms increments).
MF	FM or MFM Mode	If MF is low, FM mode is selected; if it is high, MFM mode is selected only if MFM is implemented.
MT	Multi-Track	If MT is high, a multi-track operation is to be performed. (A cylinder under both HD0 and HD1 will be read or written.)
N	Number	N stands for the number of data bytes written in a sector.

Symbol Descriptions (Part 1 of 2)

Symbol	Name	Description
NCN	New Cylinder Number	NCN stands for a new cylinder number, which is going to be reached as a result of the seek operation. (Desired position of the head.)
ND	Non-DMA Mode	ND stands for operation in the non-DMA mode.
PCN	Present Cylinder Number	PCN stands for cylinder number at the completion of sense-interrupt-status command indicating the position of the head at present time.
R	Record	R stands for the sector number, which will be read or written.
R/W	Read/Write	R/W stands for either read (R) or write (W) signal.
SC	Sector	SC indicates the number of sectors per cylinder.
SK	Skip	SK stands for skip deleted-data address mark.
SRT	Step Rate Time	SRT stands for the stepping rate for the FDD (2 to 32 ms in 2-ms increments).
ST 0 ST 1 ST 2 ST 3	Status 0 Status 1 Status 2 Status 3	ST 0-3 stand for one of four registers that store the status information after a command has been executed. This information is available during the result phase after command execution. These registers should not be confused with the main status register (selected by A0 =0). ST 0-3 may be read only after a command has been executed and contain information relevant to that particular command.
STP	Scan Test	During a scan operation, if STP =1, the data in contiguous sectors is compared byte-by-byte with data sent from the processor (or DMA), and if STP =2, then alternate sectors are read and compared.
US0, US1	Unit Select	US stands for a selected drive number encoded the same as bits 0 and 1 of the digital output register (DOR).

Symbol Descriptions (Part 2 of 2)

Command Summary

In the following table, 0 indicates "logical 0" for that bit, 1 means "logical 1," and X means "don't care."

Phase	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Remarks
					Read Data					
Command	W	MT	MF	SK	0	0	1	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information prior to command execution.
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data transfer between the FDD and main system.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					Sector ID information after command execution.
	R				H					
	R				R					
	R				N					
					Read Deleted Data					
Command	W	MT	MF	SK	0	1	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information prior to command execution.
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data transfer between the FDD and main system.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					Sector ID information after command execution.
	R				H					
	R				R					
	R				N					

Phase	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Remarks
		Write Data								
Command	W	MT	MF	0	0	0	1	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	C								Sector ID information to command execution.
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	DTL								
Execution										Data transfer between the main system and FDD.
Result	R	ST 0								Status information after command execution.
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after command execution.
	R	H								
	R	R								
	R	N								
		Write Deleted Data								
Command	W	MT	MF	0	0	1	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	C								Sector ID information prior to command execution.
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	DTL								
Execution										Data transfer between FDD and main system.
Result	R	ST 0								Status ID information after command execution.
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after command execution.
	R	H								
	R	R								
	R	N								

Phase	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Remarks
					Read a Track					
Command	W	0	MF	SK	0	0	0	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information prior to command execution.
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
	W				DTL					
Execution										Data transfer between the FDD and main system. FDC reads all of cylinder's contents from index hole to EOT.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					Sector ID information after command execution.
	R				H					
	R				R					
	R				N					
					Read ID					
Command	W	0	MF	0	0	1	0	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
Execution										The first correct ID information on the cylinder is stored in data register.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					Sector ID information during execution phase.
	R				H					
	R				R					
	R				N					

Hardware

Phase	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Remarks
					Format a Track					
Command	W	0	MF	0	0	1	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				N					Bytes/Sector
	W				SC					Sector/Track
	W				GPL					Gap 3
	W				D					filler byte.
Execution										FDC formats an entire cylinder.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					In this case, the ID information has no meaning.
	R				H					
	R				R					
	R				N					
					Scan Equal					
Command	W	MT	MF	SK	1	0	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					Sector ID information prior to command execution.
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
	W				STP					
Execution										Data compared between the FDD and the main system.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					Sector ID information after Command execution.
	R				H					
	R				R					
	R				N					

Phase	R/W	Data Bus D7	D6	D5	D4	D3	D2	D1	D0	Remarks
		Scan Low or Equal								
Command	W	MT	MF	SK	1	1	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	C								Sector ID information prior to command execution.
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	STP								
Execution										Data compared between the FDD and main system.
Result	R	ST 0								Status information after command execution.
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after command execution.
	R	H								
	R	R								
	R	N								
		Scan High or Equal								
Command	W	MT	MF	SK	1	1	1	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	C								Sector ID information prior to command execution.
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	STP								
Execution										Data compared between the FDD and main system.
Result	R	ST 0								Status information after command execution.
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after command execution.
	R	H								
	R	R								
	R	N								

Phase	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Remarks
		Recalibrate								
Command	W	0	0	0	0	0	1	1	1	Command Codes
	W	X	X	X	X	X	0	US1	US0	
Execution										Head retracted to track 0
No Result Phase										
		Sense Interrupt Status								
Command	W	0	0	0	0	1	0	0	0	Command Codes
Result	R	ST 0								Status information at the end of seek operation about the FDC
	R	PCN								
		Specify								
Command	W	0	0	0	0	0	0	1	1	Command Codes
	W	SRT				HUT				
	W	HLT							ND	
No Result Phase										
		Sense Drive Status								
Command	W	0	0	0	0	0	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
Result	R	ST 3								Status information about FDD.
		Seek								
Command	W	0	0	0	0	1	1	1	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	NCN								
Execution										Head is positioned over proper cylinder on diskette.
No Result Phase										
		Invalid								
Command	W	Invalid Codes								Invalid command codes (NoOp - FDC goes into standy state).
Result	R	ST 0								ST 0 = 80.

Bit			
No.	Name	Symbol	Description
D7 D6	Interrupt Code	IC	D7 = 0 and D6 = 0 Normal termination of command (NT). Command was completed and properly executed. D7 = 0 and D6 = 1 Abnormal termination of command (AT). Execution of command was started, but was not successfully completed. D7 = 1 and D6 = 0 Invalid command issue (IC). Command that was issued was never started. D7 = 1 and D6 = 1 Abnormal termination because, during command execution, the ready signal from FDD changed state.
D5	Seek End	SE	When the FDC completes the seek command, this flag is set to 1 (high).
D4	Equipment Check	EC	If a fault signal is received from the FDD, or if the track 0 signal fails to occur after 77 step pulses (recalibrate command), then this flag is set.
D3	Not Ready	NR	When the FDD is in the not-ready state and a read or write command is issued, this flag is set. If a read or write command is issued to side 1 of a single-sided drive, then this flag is set.
D2	Head Address	HD	This flag is used to indicate the state of the head at interrupt.
D1 D0	Unit Select 1 Unit Select 0	US 1 US 0	These flags are used to indicate a drive unit number at interrupt.

Command Status Register 0

Bit			Description
No.	**Name**	**Symbol**	
D7	End of Cylinder	EN	When the FDC tries to access a sector beyond the final sector of a cylinder, this flag is set.
D6	—	—	Not used. This bit is always 0 (low).
D5	Data Error	DE	When the FDC detects a CRC error in either the ID field or the data field, this flag is set.
D4	Over Run	OR	If the FDC is not serviced by the main system during data transfers within a certain time interval, this flag is set.
D3	—	—	Not used. This bit is always 0 (low).
D2	No Data	ND	During execution of a read data, write deleted data, or scan command, if the FDC cannot find the sector specified in the ID register, this flag is set. During execution of the read ID command, if the FDC cannot read the ID field without an error, then this flag is set. During the execution of the read a cylinder command, if the starting sector cannot be found, then this flag is set.
D1	Not Writable	NW	During execution of a write data, write deleted data, or format-a-cylinder command, if the FDC detects a write-protect signal from the FDD, then this flag is set.
D0	Missing Address Mark	MA	If the FDC cannot detect the ID address mark, this flag is set. Also, at the same time, the MD (missing address mark in the data field) of status register 2 is set.

Command Status Register 1

Bit			
No.	Name	Symbol	Description
D7	—	—	Not used. This bit is always 0 (low).
D6	Control Mark	CM	During execution of the read data or scan command, if the FDC encounters a sector that contains a deleted data address mark, this flag is set.
D5	Data Error in Data Field	DD	If the FDC detects a CRC error in the data, then this flag is set.
D4	Wrong Cylinder	WC	This bit is related to the ND bit, and when the contents of C on the medium are different from that stored in the ID register, this flag is set.
D3	Scan Equal Hit	SH	During execution of the scan command, if the condition of "equal" is satisfied, this flag is set.
D2	Scan Not Satisfied	SN	During execution of the scan command, if the FDC cannot find a sector on the cylinder that meets the condition, then this flag is set.
D1	Bad Cylinder	BC	This bit is related to the ND bit, and when the contents of C on the medium are different from that stored in the ID register, and the contents of C is FF, then this flag is set.
D0	Missing Address Mark in Data Field	MD	When data is read from the medium, if the FDC cannot find a data address mark or deleted data address mark, then this flag is set.

Command Status Register 2

Bit No.	Bit Name	Bit Symbol	Description
D7	Fault	FT	This bit is the status of the fault signal from the FDD.
D6	Write Protected	WP	This bit is the status of the write-protected signal from the FDD.
D5	Ready	RY	This bit is the status of the ready signal from the FDD.
D4	Track 0	T0	This bit is the status of the track 0 signal from the FDD.
D3	Two Side	TS	This bit is the status of the two-side signal from the FDD.
D2	Head Address	HD	This bit is the status of the side-select signal from the FDD.
D1	Unit Select 1	US 1	This bit is the status of the unit-select-1 signal from the FDD.
D0	Unit Select 0	US 0	This bit is the status of the unit-select-0 signal from the FDD.

Command Status Register 3

Programming Summary

FDC Data Register		I/O Address Hex 3F5	
FDC Main Status Register		I/O Address Hex 3F4	
Digital Output Register		I/O Address Hex 3F2	
Bit 0	Drive	00: DR #A	10: DR #C
1	Select	01: DR #B	11: DR #D
2	Not FDC Reset		
3	Enable INT & DMA Requests		
4	Drive A Motor Enable		
5	Drive B Motor Enable		
6	Drive C Motor Enable		
7	Drive D Motor Enable		

All bits cleared with channel reset.

DPC Registers

FDC Constants (in hex)

N:	02	GPL Format:	05
SC:	08	GPL R/W:	2A
HUT:	F	HLT:	01
SRT:	C	(6 ms track-to-track)	

Drive Constants

Head Load	35 ms
Head Settle	15 ms
Motor Start	250 ms

Comments

- Head loads with drive select, wait HD load before R/W.
- Following access, wait HD settle time before R/W.
- Drive motors should be off when not in use. Only A or B and C or D may run simultaneously. Wait motor start time before R/W.
- Motor must be on for drive to be selected.
- Data errors can occur while using a home television as the system display. Locating the TV too close to the diskette area can cause this to occur. To correct the problem, move the TV away from, or to the opposite side of the system unit.

System I/O Channel Interface

All signals are TTL-compatible:

Most Positive Up Level	5.5 Vdc
Least Positive Up Level	2.7 Vdc
Most Positive Down Level	0.5 Vdc
Least Positive Down Level	−0.5 Vdc

The following lines are used by this adapter.

+D0-7	(Bidirectional, load: 1 74LS, driver: 74LS 3-state). These eight lines form a bus by which all commands, status, and data are transferred. Bit 0 is the low-order bit.
+A0-9	(Adapter input, load: 1 74LS) These ten lines form an address bus by which a register is selected to receive or supply the byte transferred through lines D0-7. Bit 0 is the low-order bit.
+AEN	(Adapter input, load: 1 74LS) The content of lines A0-9 is ignored if this line is active.
−IOW	(Adapter input, load: 1 74LS) The content of lines D0-7 is stored in the register addressed by lines A0-9 or DACK2 at the trailing edge of this signal.
−IOR	(Adapter input, load: 1 74LS) The content of the register addressed by lines A0-9 or DACK2 is gated onto lines D0-7 when this line is active.
−DACK2	(Adapter input, load: 2 74LS) This line being active degates output DRQ2, selects the FDC data register as the source/destination of bus D0-7, and indirectly gates T/C to IRQ6.
+T/C	(Adapter input, load: 4 74LS) This line and DACK2 being active indicates that the byte of data for which the DMA count was initialized is now being transferred.
+RESET	(Adapter input, load: 1 74LS) An up level aborts any operation in process and clears the digital output register (DOR).

+DRQ2 (Adapter output, driver: 74LS 3-state)
This line is made active when the attachment is ready to transfer a byte of data to or from main storage. The line is made inactive by DACK2 becoming active or an I/O read of the FDC data register.

+IRQ6 (Adapter output, driver: 74LS 3-state)
This line is made active when the FDC has completed an operation. It results in an interrupt to a routine which should examine the FDC result bytes to reset the line and determine the ending condition.

Drive A and B Interface

All signals are TTL-compatible:

Most Positive Up Level	5.5 Vdc
Least Positive Up Level	2.4 Vdc
Most Positive Down Level	0.4 Vdc
Least Positive Down Level	−0.5 Vdc

All adapter outputs are driven by open-collector gates. The drive(s) must provide termination networks to Vcc (except motor enable, which has a 2000-ohm resistor to Vcc).

Each adapter input is terminated with a 150-ohm resistor to Vcc.

Adapter Outputs

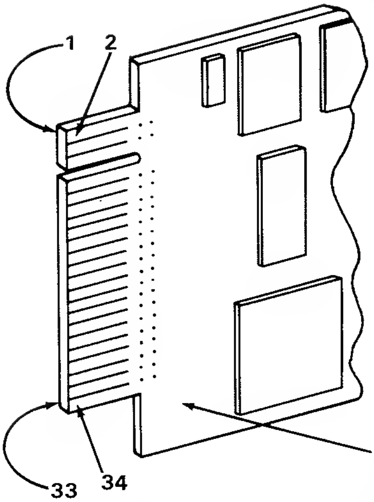
−Drive Select A and B (Driver: 7438)
These two lines are used by drives A and B to degate all drivers to the adapter and receivers from the attachment (except motor enable) when the line associated with a drive is inactive.

—Motor Enable A and B	(Driver: 7438) The drive associated with each of these lines must control its spindle motor such that it starts when the line becomes active and stops when the line becomes inactive.
—Step	(Driver: 7438) The selected drive moves the read/write head one cylinder in or out per the direction line for each pulse present on this line.
—Direction	(Driver: 7438) For each recognized pulse of the step line, the read/write head moves one cylinder toward the spindle if this line is active, and away from the spindle if inactive.
—Head Select	(Driver: 7438) Head 1 (upper head) will be selected when this line is active (low).
—Write Data	(Driver: 7438) For each inactive to active transition of this line while write enable is active, the selected drive causes a flux change to be stored on the diskette.
—Write Enable	(Driver: 7438) The drive disables write current in the head unless this line is active.

Adapter Inputs

—Index	The selected drive supplies one pulse per diskette revolution on this line.
—Write Protect	The selected drive makes this line active if a write-protected diskette is mounted in the drive.

−Track 0	The selected drive makes this line active if the read/write head is over track 0.
−Read Data	The selected drive supplies a pulse on this line for each flux change encountered on the diskette.

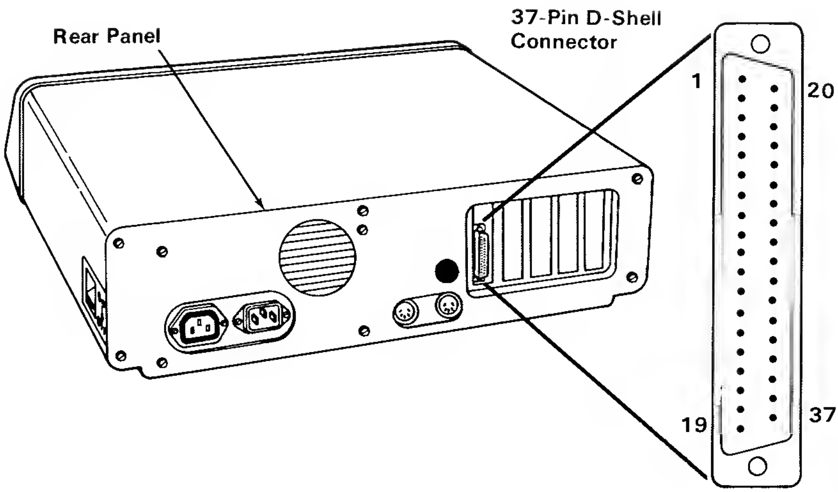


Note: Lands 1-33 (odd numbers) are on the back of the board. Lands 2-34 (even numbers) are on the front, or component side.

Diskette Drives ↔ Drive Adapter

At Standard TTL Levels	Land Number
Ground-Odd Numbers	1-33
Unused	2,4,6
Index	8
Motor Enable A	10
Drive Select B	12
Drive Select A	14
Motor Enable B	16
Direction (Stepper Motor)	18
Step Pulse	20
Write Data	22
Write Enable	24
Track 0	26
Write Protect	28
Read Data	30
Select Head 1	32
Unused	34

Connector Specifications (Part 1 of 2)



External Drives ↔ Drive Adapter: At Standard TTL Levels	Pin Number
Unused	1-5
Index	6
Motor Enable C	7
Drive Select D	8
Drive Select C	9
Motor Enable D	10
Direction (Stepper Motor)	11
Step Pulse	12
Write Data	13
Write Enable	14
Track 0	15
Write Protect	16
Read Data	17
Select Head 1	18
Ground	20-37

Connector Specifications (Part 2 of 2)

IBM 5-1/4" Diskette Drive

The system unit has space and power for one or two 5-1/4 inch diskette drives. A drive can be single-sided or double-sided with 40 tracks for each side, is fully self-contained, and consists of a spindle drive system, a read positioning system, and a read/write/erase system.

The diskette drive uses modified frequency modulation (MFM) to read and write digital data, with a track-to-track access time of 6 milliseconds.

To load a diskette, the operator raises the latch at the front of the diskette drive and inserts the diskette into the slot. Plastic guides in the slot ensure the diskette is in the correct position. Closing the latch centers the diskette and clamps it to the drive hub. After 250 milliseconds, the servo-controlled dc drive motor starts and drives the hub at a constant speed of 300 rpm. The head positioning system, which consists of a 4-phase stepper-motor and band assembly with its associated electronics, moves the magnetic head so it comes in contact with the desired track of the diskette. The stepper-motor and band assembly uses one-step rotation to cause a one-track linear movement of the magnetic head. No operator intervention is required during normal operation. During a write operation, a 0.013-inch (0.33 millimeter) data track is recorded, then tunnel-erased to 0.012 inch (0.030 millimeter). If the diskette is write-protected, a write-protect sensor disables the drive's circuitry, and an appropriate signal is sent to the interface.

Data is read from the diskette by the data-recovery circuitry, which consists of a low-level read amplifier, differentiator, zero-crossing detector, and digitizing circuits. All data decoding is done by an adapter card.

The diskette drive also has the following sensor systems:

1. The track 00 switch, which senses when the head/carriage assembly is at track 00.

2. The index sensor, which consists of an LED light source and phototransistor. This sensor is positioned so that when an index hole is detected, a digital signal is generated.

3. The write-protect sensor disables the diskette drive's electronics whenever a write-protect tab is applied to the diskette.

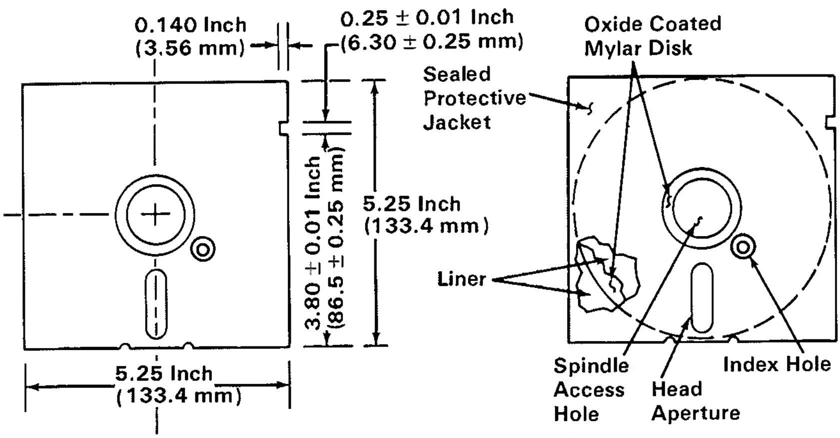
For interface information, refer to "IBM 5-1/4" Diskette Drive Adapter" earlier in this section.

Media	Industry-compatible 5-1/4 inch diskette
Tracks per inch	48
Number of tracks	40
Dimensions	
Height	3.38 inches (85.85 mm)
Width	5.87 inches (149.10 mm)
Depth	8.00 inches (203.2 mm)
Weight	4.50 pounds (2.04 kg)
Temperature (Exclusive of media)	
Operating	50°F to 112°F (10°C to 44°C)
Non operating	-40°F to 140°F (-40°C to 60°C)
Relative humidity (Exclusive of media)	
Operating	20% to 80% (non condensing)
Non operating	5% to 95% (non condensing)
Seek Time	6 ms track-to-track
Head Settling Time	15 ms (last track addressed)
Error Rate	1 per 10^9 (recoverable) 1 per 10^{12} (non recoverable) 1 per 10^6 (seeks)
Head Life	20,000 hours (normal use)
Media Life	3.0 x 10^6 passes per track
Disk Speed	300 rpm +/- 1.5% (long term)
Instantaneous Speed Variation	+/- 3.0%
Start/Stop Time	250 ms (maximum)
Transfer Rate	250K bits/sec
Recording Mode	MFM
Power	+12 Vdc +/- 0.6 V, 900 mA average +5 Vdc +/- 0.25 V, 600 mA average

Mechanical and Electrical Specifications

Diskettes

The IBM 5-1/4" Diskette Drive uses a standard 5.25-inch (133.4-millimeter) diskette. For programming considerations, single-sided, double-density, soft-sectored diskettes are used for single-sided drives. Double-sided drives use double-sided, double-density, soft-sectored diskettes. The figure below is a simplified drawing of the diskette used with the diskette drive. This recording medium is a flexible magnetic disk enclosed in a protective jacket. The protected disk, free to rotate within the jacket, is continuously cleaned by the soft fabric lining of the jacket during normal operation. Read/write/erase head access is made through an opening in the jacket. Openings for the drive hub and diskette index hole are also provided.



Recording Medium

Notes:

IBM Fixed Disk Drive Adapter

The fixed disk drive adapter attaches to one or two fixed disk drive units, through an internal daisy-chained flat cable (data/control cable). Each system supports a maximum of one fixed disk drive adapter and two fixed disk drives.

The adapter is buffered on the I/O bus and uses the system board direct memory access (DMA) for record data transfers. An interrupt level also is used to indicate operation completion and status conditions that require processor attention.

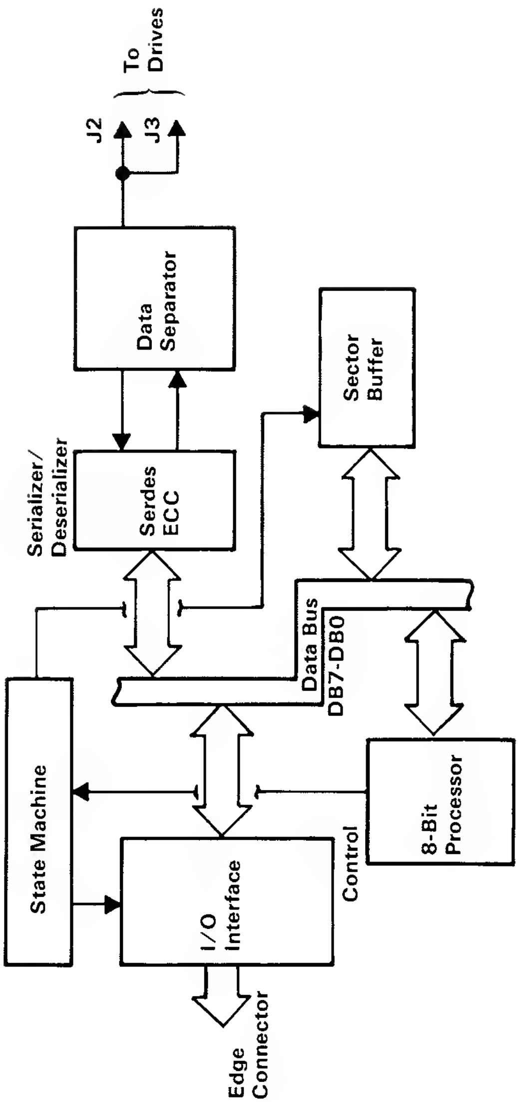
The fixed disk drive adapter provides automatic 11-bit burst error detection and correction in the form of 32-bit error checking and correction (ECC).

The device level control for the fixed disk drive adapter is contained on a ROM module on the adapter. A listing of this device level control can be found in "Appendix A: ROM BIOS Listings."

WARNING: The last cylinder on the fixed disk drive is reserved for diagnostic use. Diagnostic write tests will destroy any data on this cylinder.

Fixed Disk Controller

The disk controller has two registers that may be accessed by the main system processor: a status register and a data register. The 8-bit status register contains the status information of the disk controller, and can be accessed at any time. The 8-bit data register (actually consisting of several registers in a stack with only one register presented to the data bus) stores data, commands, parameters, and provides the disk controller's status information. Data bytes are read from, or written to the data register in order to program or obtain the results after a particular command. The status register is a read-only register, and is used to help the transfer of data between the processor and the disk controller. The controller-select pulse is generated by writing to port address hex 322.



Fixed Disk Drive Adapter Block Diagram

Programming Considerations

Status Register

At the end of all commands from the system board, the disk controller returns a completion status byte back to the system board. This byte informs the system unit if an error occurred during the execution of the command. The following shows the format of this byte.

Bit	7	6	5	4	3	2	1	0
	0	0	d	0	0	0	e	0

Bits 0, 1, 2, 3, 4, 6, 7 These bits are set to zero.

Bit 1 When set, this bit shows an error has occurred during command execution.

Bit 5 This bit shows the logical unit number of the drive.

If the interrupts are enabled, the controller sends an interrupt when it is ready to transfer the status byte. Busy from the disk controller is unasserted when the byte is transferred to complete the command.

Sense Bytes

If the status register receives an error (bit 1 is set), then the disk controller requests four bytes of sense data. The format for the four bytes is as follows:

Bits	7	6	5	4	3	2	1	0
Byte 0	Address Valid	0	Error Type		Error Code			
Byte 1	0	0	d	Head Number				
Byte 2	Cylinder High			Sector Number				
Byte 3	Cylinder Low							

Remarks

d = drive

Byte 0	Bits 0, 1, 2, 3	Error code.
Byte 0	Bits 4, 5	Error type.
Byte 0	Bit 6	Set to 0 (spare).
Byte 0	Bit 7	The address valid bit. Set only when the previous command required a disk address, in which case it is returned as a 1; otherwise, it is a 0.

The following disk controller tables list the error types and error codes found in byte 0:

	Error Type	Error Code	
Bits	**5 4**	**3 2 1 0**	**Description**
	0 0	0 0 0 0	The controller did not detect any error during the execution of the previous operation.
	0 0	0 0 0 1	The controller did not detect an index signal from the drive.
	0 0	0 0 1 0	The controller did not get a seek-complete signal from the drive after a seek operation (for all non-buffered step seeks).
	0 0	0 0 1 1	The controller detected a write fault from the drive during the last operation.
	0 0	0 1 0 0	After the controller selected the drive, the drive did not respond with a ready signal.
	0 0	0 1 0 1	Not used.
	0 0	0 1 1 0	After stepping the maximum number of cylinders, the controller did not receive the track 00 signal from the drive.
	0 0	0 1 1 1	Not used.
	0 0	1 0 0 0	The drive is still seeking. This status is reported by the Test Drive Ready command for an overlap seek condition when the drive has not completed the seek. No time-out is measured by the controller for the seek to complete.

	Error Type	Error Code	
Bits	5 4	3 2 1 0	Description
	0 1	0 0 0 0	ID Read Error: The controller detected an ECC error in the target ID field on the disk.
	0 1	0 0 0 1	Data Error: The controller detected an uncorrectable ECC error in the target sector during a read operation.
	0 1	0 0 1 0	Address Mark: The controller did not detect the target address mark (AM) on the disk.
	0 1	0 0 1 1	Not used.
	0 1	0 1 0 0	Sector Not Found: The controller found the correct cylinder and head, but not the target sector.
	0 1	0 1 0 1	Seek Error: The cylinder or head address (either or both) did not compare with the expected target address as a result of a seek.
	0 1	0 1 1 0	Not used.
	0 1	0 1 1 1	Not used.
	0 1	1 0 0 0	Correctable Data Error: The controller detected a correctable ECC error in the target field.
	0 1	1 0 0 1	Bad Track: The controller detected a bad track flag during the last operation. No retries are attempted on this error.

	Error Type	Error Code	
Bits	**5 4**	**3 2 1 0**	**Description**
	1 0	0 0 0 0	Invalid Command: The controller has received an invalid command from the system unit.
	1 0	0 0 0 1	Illegal Disk Address: The controller detected an address that is beyond the maximum range.

	Error Type	Error Code	
Bits	**5 4**	**3 2 1 0**	**Description**
	1 1	0 0 0 0	RAM Error: The controller detected a data error during the RAM sector-buffer diagnostic test.
	1 1	0 0 0 1	Program Memory Checksum Error: During this internal diagnostic test, the controller detected a program-memory checksum error.
	1 1	0 0 1 0	ECC Polynominal Error: During the controller's internal diagnostic tests, the hardware ECC generator failed its test.

Data Register

The processor specifies the operation by sending the 6-byte device control block (DCB) to the controller. The figure below shows the composition of the DCB, and defines the bytes that make up the DCB.

Bit	7	6	5	4	3	2	1	0
Byte 0	Command Class			Opcode				
Byte 1	0	0	d	Head Number				
Byte 2	Cylinder High		Sector Number					
Byte 3	Cylinder Low							
Byte 4	Interleave or Block Count							
Byte 5	Control Field							

Byte 0 – Bits 7, 6, and 5 identify the class of the command.
Bits 4 through 0 contain the Opcode command.

Byte 1 – Bit 5 identifies the drive number.
Bits 4 through 0 contain the disk head number to be selected.
Bits 6 and 7 are not used.

Byte 2 – Bits 6 and 7 contain the two most significant bits of the cylinder number.
Bits 0 through 5 contain the sector number.

Byte 3 – Bits 0 through 7 are the eight least significant bits of the cylinder number.

Byte 4 – Bits 0 through 7 specify the interleave or block count.

Byte 5 – Bits 0 through 7 contain the control field.

Control Byte

Byte 5 is the control field of the DCB and allows the user to select options for several types of disk drives. The format of this byte is as follows:

Bits	7	6	5	4	3	2	1	0
	r	a	0	0	0	s	s	s

Remarks

r = retries
s = step option
a = retry option on data ECC error

Bit 7 Disables the four retries by the controller on all disk-access commands. Set this bit only during the evaluation of the performance of a disk drive.

Bit 6 If set to 0 during read commands, a reread is attempted when an ECC error occurs. If no error occurs during reread, the command will complete with no error status. If this bit is set to 1, no reread is attempted.

Bits 5, 4, 3 Set to 0.

Bits 2, 1, 0 These bits define the type of drive and select the step option. See the following figure.

Bits 2, 1, 0	
0 0 0	This drive is not specified and defaults to 3 milliseconds per step.
0 0 1	N/A
0 1 0	N/A
0 1 1	N/A
1 0 0	200 microseconds per step.
1 0 1	70 microseconds per step (specified by BIOS).
1 1 0	3 milliseconds per step.
1 1 1	3 milliseconds per step.

Command Summary

<table>
<tr><th>Command</th><th>Data Control Block</th><th>Remarks</th></tr>
<tr><td>Test Drive Ready (Class 0, Opcode 00)</td><td><table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr>
</table></td><td>d = drive (0 or 1)
x = don't care
Bytes 2, 3, 4, 5 = don't care</td></tr>
<tr><td>Recalibrate (Class 0, Opcode 01)</td><td><table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr>
<tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr>
</table></td><td>d = drive (0 or 1)
x = don't care
r = retries
s = Step Option
Bytes 2, 3, 4 = don't care
ch = cylinder high</td></tr>
<tr><td>Reserved (Class 0, Opcode 02)</td><td></td><td>This Opcode is not used.</td></tr>
<tr><td>Request Sense Status (Class 0, Opcode 03)</td><td><table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr>
</table></td><td>d = drive (0 or 1)
x = don't care
Bytes 2, 3, 4, 5 = don't care</td></tr>
<tr><td>Format Drive (Class 0, Opcode 04)</td><td><table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr>
<tr><td>Byte 2</td><td colspan="2">ch</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr>
<tr><td>Byte 4</td><td>0</td><td>0</td><td>0</td><td colspan="5">Interleave</td></tr>
<tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr>
</table></td><td>d = drive (0 or 1)
r = retries
s = step option
ch = cylinder high

Interleave: 1 to 16 for 512-byte sectors.</td></tr>
<tr><td>Ready Verify (Class 0, Opcode 05)</td><td><table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr>
<tr><td>Byte 2</td><td colspan="2">ch</td><td colspan="6">Sector Number</td></tr>
<tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr>
<tr><td>Byte 4</td><td colspan="8">Block Count</td></tr>
<tr><td>Byte 5</td><td>r</td><td>a</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr>
</table></td><td>d = drive (0 or 1)
r = retries
s = step option
a = retry option on data ECC
ch = cylinder high</td></tr>
</table>

Command	Data Control Block	Remarks

Format Track (Class 0, Opcode 06)

Bit	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	0	1	1	0
Byte 1	0	0	d	Head Number				
Byte 2	ch		0	0	0	0	0	0
Byte 3	Cylinder Low							
Byte 4	0	0	0	Interleave				
Byte 5	r	0	0	0	0	s	s	s

d = drive (0 or 1)
r = retries
s = step option
ch =cylinder high

Interleave: 1 to 16 for 512-byte sectors

Format Bad Track (Class 0, Opcode 07)

Bit	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	0	1	1	1
Byte 1	0	0	d	Head Number				
Byte 2	ch		0	0	0	0	0	0
Byte 3	Cylinder Low							
Byte 4	0	0	0	Interleave				
Byte 5	r	0	0	0	0	s	s	s

d = drive (0 or 1)
r = retries
s = step option
ch = cylinder high

Interleave: 1 to 16 for 512-byte sectors

Read (Class 0, Opcode 08)

Bit	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	1	0	0	0
Byte 1	0	0	d	Head Number				
Byte 2	ch		Sector Number					
Byte 3	Cylinder Low							
Byte 5	r	a	0	0	0	s	s	s

d = drive (0 or 1)
r = retries
a = retry option on data ECC error
s = step option
ch =cylinder high

Reserved (Class 0, (Opcode 09)

This Opcode is not used

Write (Class 0, Opcode 0A)

Bit	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	1	0	1	0
Byte 1	0	0	d	Head Number				
Byte 2	ch		Sector Number					
Byte 3	Cylinder Low							
Byte 4	Block Count							
Byte 5	r	0	0	0	0	s	s	s

d = drive (0 or 1)
r = retries
s = step option
ch = cylinder high

Seek (Class 0, Opcode 0B)

Bit	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	1	0	1	1
Byte 1	0	0	d	Head Number				
Byte 2	ch		0	0	0	0	0	0
Byte 3	Cylinder Low							
Byte 4	x	x	x	x	x	x	x	x
Byte 5	r	0	0	0	0	s	s	s

d = drive (0 or 1)
r = retries
s = step option
x = don't care
ch = cylinder high

Command	Data Control Block	Remarks
Initialize Drive Characteristics* (Class 0, Opcode 0C)	Bit: 7 6 5 4 3 2 1 0 Byte 0: 0 0 0 | 0 1 1 0 0	Bytes 1, 2, 3, 4, 5 = don't care
Read ECC Burst Error Length (Class 0, Opcode 0D)	Bit: 7 6 5 4 3 2 1 0 Byte 0: 0 0 0 | 0 1 1 0 1	Bytes 1, 2, 3, 4, 5 = don't care
Read Data from Sector Buffer (Class 0, Opcode 0E)	Bit: 7 6 5 4 3 2 1 0 Byte 0: 0 0 0 | 0 1 1 1 0	Bytes 1, 2, 3, 4, 5 = don't care
Write Data to Sector Buffer (Class 0, Opcode 0F)	Bit: 7 6 5 4 3 2 1 0 Byte 0: 0 0 0 | 0 1 1 1 1	Bytes 1, 2, 3, 4, 5 = don't care
RAM Diagnostic (Class 7, Opcode 00)	Bit: 7 6 5 4 3 2 1 0 Byte 0: 1 1 1 | 0 0 0 0 0	Bytes 1, 2, 3, 4, 5 = don't care
Reserved (Class 7, Opcode 01)		This Opcode is not used
Reserved (Class 7, Opcode 02)		This Opcode is not used

*Initialize Drive Characteristics: The DCB must be followed by eight additional bytes.

Maximum number of cylinders	(2 bytes)
Maximum number of heads	(1 byte)
Start reduced write current cylinder	(2 bytes)
Start write precompensation cylinder	(2 bytes)
Maximum ECC data burst length	(1 byte)

<table>
<tr><th>Command</th><th>Data Control Block</th><th>Remarks</th></tr>
<tr><td>Drive
Diagnostic
(Class 7,
Opcode 03)</td><td>
<table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr>
<tr><td>Byte 2</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr>
<tr><td>Byte 3</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr>
<tr><td>Byte 4</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr>
<tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr>
</table>
</td><td>d = drive (0 or 1)
s = step option
r = retries
x = don't care</td></tr>
<tr><td>Controller
Internal
Diagnostics
(Class 7,
Opcode 04)</td><td>
<table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
</table>
</td><td>Bytes 1, 2, 3, 4, 5 = don't care</td></tr>
<tr><td>Read Long*
(Class 7,
Opcode 05)</td><td>
<table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr>
<tr><td>Byte 2</td><td colspan="2">ch</td><td colspan="6">Sector Number</td></tr>
<tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr>
<tr><td>Byte 4</td><td colspan="8">Block Count</td></tr>
<tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr>
</table>
</td><td>d = drive (0 or 1)
s = step option
r = retries
ch = cylinder high</td></tr>
<tr><td>Write Long**
(Class 7,
Opcode 06)</td><td>
<table>
<tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Byte 0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr>
<tr><td>Byte 2</td><td colspan="2">ch</td><td colspan="6">Sector Number</td></tr>
<tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr>
<tr><td>Byte 4</td><td colspan="8">Block Count</td></tr>
<tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr>
</table>
</td><td>d = drive (0 or 1)
s = step option
r = retries
ch = cylinder high</td></tr>
</table>

*Returns 512 bytes plus 4 bytes of ECC data per sector.

**Requires 512 bytes plus 4 bytes of ECC data per sector.

Programming Summary

The two least-significant bits of the address bus are sent to the system board's I/O port decoder, which has two sections. One section is enabled by the I/O read signal (−IOR) and the other by the I/O write signal (−IOW). The result is a total of four read/write ports assigned to the disk controller board.

The address enable signal (AEN) is asserted by the system board when DMA is controlling data transfer. When AEN is asserted, the I/O port decoder is disabled.

The following figure is a table of the four read/write ports:

R/W	Port Address	Function
Read	320	Read data (from controller to system unit).
Write	320	Write data (from system unit to controller).
Read	321	Read controller hardware status.
Write	321	Controller reset.
Read	322	Reserved.
Write	322	Generate controller-select pulse.
Read	323	Not used.
Write	323	Write pattern to DMA and interrupt mask register.

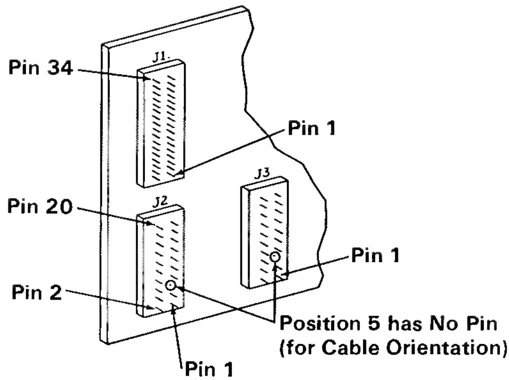
System I/O Channel Interface

The following lines are used by the disk controller:

A0-A19	Positive true 20-bit address. The least-significant 10 bits contain the I/O address within the range of hex 320 to hex 323 when an I/O read or write is executed by the system unit. The full 20 bits are decoded to address the read-only memory (ROM) between the addresses of hex C8000 and C9FFF.
D0-D7	Positive 8-bit data bus over which data and status information is passed between the system board and the controller.
$\overline{\text{IOR}}$	Negative true signal that is asserted when the system board reads status or data from the controller under either programmed I/O or DMA control.
$\overline{\text{IOW}}$	Negative true signal that is asserted when the system board sends a command or data to the controller under either programmed I/O or DMA control.
AEN	Positive true signal that is asserted when the DMA in the system board is generating the I/O Read (−IOR) or I/O Write (−IOW) signals and has control of the address and data buses.
RESET	Positive true signal that forces the disk controller to its initial power-up condition.
IRQ 5	Positive true interrupt request signal that is asserted by the controller, when enabled to interrupt the system board on the return ending status byte from the controller.

DRQ 3 — Positive-true DMA-request signal that is asserted by the controller when data is available for transfer to or from the controller under DMA control. This signal remains active until the system board's DMA channel activates the DMA-acknowledge signal (−DACK 3) in response.

$\overline{\text{DACK 3}}$ — This signal is true when negative, and is generated by the system board DMA channel in response to a DMA request (DRQ 3).



Disk Drive Connector J1 — Disk Adapter Connector J1

Signal	Pin Number
Ground - Odd Numbers	1-33
Reserved	4, 16, 30, 32
-Reduced Write Current	2
-Write Gate	6
-Seek Complete	8
-Track 00	10
-Write Fault	12
-Head Select 2^0	14
-Head Select 2^1	18
-Index	20
-Ready	22
-Step	24
-Drive Select 1	26
-Drive Select 2	28
-Direction In	34

Disk Drive Connector J2 or J3 — Disk Adapter Connector J2 or J3

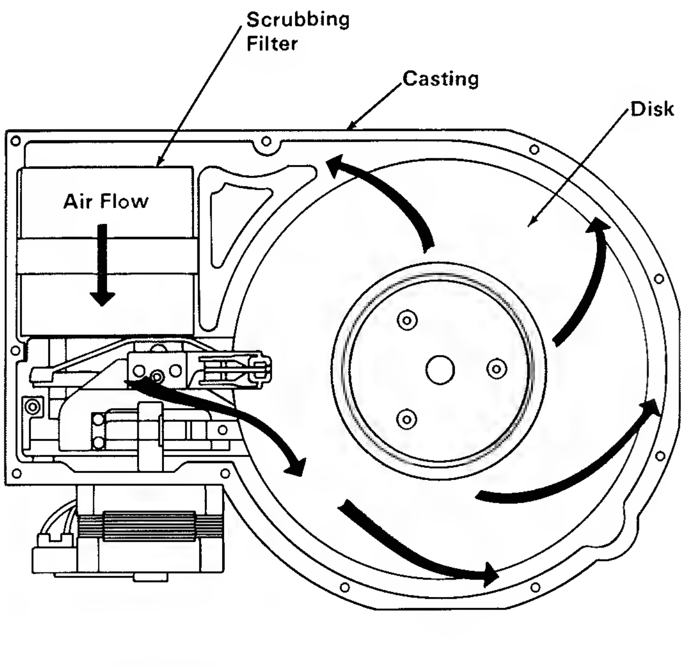
Signal	Pin Number
Ground	2, 4, 6, 8, 12, 16, 20
Drive Select	1
Reserved	3, 7
Spare	9, 10, 5 (No Pin)
Ground	11
MFM Write Data	13
-MFM Write Data	14
Ground	15
MFM Read Data	17
-MFM Read Data	18
Ground	19

Fixed Disk Adapter Interface Specifications

IBM 10MB Fixed Disk Drive

The disk drive is a random-access storage device that uses two non-removable 5-1/4 inch disks for storage. Each disk surface employs one movable head to service 306 cylinders. The total formatted capacity of the four heads and surfaces is 10 megabytes (17 sectors per track with 512 bytes per sector and a total of 1224 tracks).

An impact-resistant enclosure provides mechanical and contamination protection for the heads, actuator, and disks. A self-contained recirculating system supplies clean air through a 0.3-micron filter. Thermal isolation of the stepper and spindle motor assemblies from the disk enclosure results in a very low temperature rise within the enclosure. This isolation provides a greater off-track margin and the ability to perform read and write operations immediately after power-up with no thermal stabilization delay.



Media	Rigid media disk
Number of Tracks	1224
Track Density	345 tracks per inch
Dimensions	
Height	3.25 inches (82.55 mm)
Width	5.75 inches (146.05 mm)
Depth	8.0 inches (203.2 mm)
Weight	4.6 lb (2.08 kg)
Temperature	
Operating	40°F to 122°F (4°C to 50°C)
Non operating	-40°F to 140°F (-40°C to 60°C)
Relative Humidity	
Operating	8% to 80% (non condensing)
Maximum Wet Bulb	78°F (26°C)
Shock	
Operating	10 Gs
Non operating	20 Gs
Access Time	3 ms track-to-track
Average Latency	8.33 ms
Error Rates	
Soft Read Errors	1 per 10^{10} bits read
Hard Read Errors	1 per 10^{12} bits read
Seek Errors	1 per 10^{6} seeks
Design Life	5-years (8,000 hours MTF)
Disk Speed	3600 rpm ±1%
Transfer Rate	5.0 M bits/sec
Recording Mode	MFM
Power	+12 Vdc ± 5% 1.8 A (4.5 A maximum) +5 Vdc ± 5% 0.7 A (1.0 A maximum)
Maximum Ripple	1% with equivalent resistive load

Mechanical and Electrical Specifications

IBM Memory Expansion Options

Three memory expansion options (32KB, 64KB, and 64/256KB) and two memory module kits (16KB and 64KB) are available for the IBM Personal Computer. Memory expansion is described in the following chart:

	Minimum Memory	Maximum Memory	Number of 16K Memory Module Kits	Number of 64K Memory Module Kits	Memory Module Type
16/64K System Board	16K	64K	1, 2, or 3		16K by 1 Bit, 16 pin
64/256K System Board	64K	256K		1, 2, or 3	64K by 1 Bit, 16 pin
64/256K Memory Option	64K	256K		1, 2, or 3	64K by 1 Bit, 16 pin
32K Memory Option	32K				16K by 1 Bit, 16 pin
64K Memory Option	64K				Stacked 32K by 1 Bit, 18 pin

The system board must be fully populated before any memory expansion options can be installed. An expansion option must be configured to reside at a sequential 32K or 64K memory address boundary within the system address space. This is done by setting the DIP switches on the option.

All memory expansion options are parity checked. If a parity error is detected, a latch is set and an I/O channel check line is activated, indicating an error to the processor.

In addition to the memory modules, the memory expansion options contain the following circuits: bus buffering, dynamic memory timing generation, address multiplexing, and card-select decode logic.

Dynamic-memory refresh timing and address generation are functions performed on the system board and made available in the I/O channel for all devices.

To allow the system to address 32K, 64K, or 64/256K memory expansion options, refer to "Appendix G: Switch Settings" for the proper memory expansion option switch settings.

Operating Characteristics

The system board operates at a frequency of 4.77 MHz, which results in a clock cycle of 210 ns.

Normally four clock cycles are required for a bus cycle so that an 840-ns memory cycle time is achieved. Memory-write and memory-read cycles both take four clock cycles, or 840 ns.

General specifications for memory used on all cards are:

	16K by 1 Bit	32K by 1 Bit	64K by 1 Bit
Access	250 ns	250 ns	200 ns
Cycle	410 ns	410 ns	345 ns

Memory Module Description

Both the 32K and the 64K options contain 18 dynamic memory modules. The 32K memory expansion option utilizes 16K by 1 bit modules, and the 64K memory expansion option utilizes 32K by 1 bit modules.

The 64/256K option has four banks of 9 pluggable sockets. Each bank will accept a 64K memory module kit, consisting of 9 (64K by 1) modules. The kits must be installed sequentially into banks 1, 2, and 3. The base 64/256K option comes with modules installed in bank 0, providing 64K of memory. One, two, or three 64K bits may be added, upgrading the option to 128K, 192K, or 256K of memory.

The 16K by 1 and the 32K by 1 modules require three voltage levels: +5 Vdc, −5 Vdc, and +12 Vdc. The 64K by 1 modules require only one voltage level of +5 Vdc. All three memory modules require 128 refresh cycles every 2 ns. Absolute maximum access times are:

	16K by 1 Bit	32K by 1 Bit	64K by 1 Bit
From $\overline{\text{RAS}}$	250 ns	250 ns	200 ns
From $\overline{\text{CAS}}$	165 ns	165 ns	115 ns

Pin	16K by 1 Bit Module (used on 32K option and 16/64K system board)	32K by 1 Bit Module (used on 64K option)	64K by 1 Bit Module (used on 64/256K option and 64/256K system board)
1	-5 Vdc	-5 Vdc	N/C
2	Data In**	Data In**	Data In***
3	-Write	-Write	-Write
4	-RAS	-RAS 0	-RAS
5	A0	-RAS 1	A0
6	A2	A0	A2
7	A1	A2	A1
8	+12 Vdc	A1	+5 Vdc
9	+5 Vdc	+12 Vdc	A7
10	A5	+5 Vdc	A5
11	A4	A5	A4
12	A3	A4	A3
13	A6	A3	A6
14	Data Out**	A6	Data Out***
15	-CAS	Data Out**	-CAS
16	GND	-CAS 1	GND
17	*	-CAS 0	*
18	*	GND	*

*16K by 1 and 64K by 1 bit modules have 16 pins.
**Data In and Data Out are tied together (three-state bus).
***Data In and Data Out are tied together on Data Bits 0-7 (three-state bus).

Memory Module Pin Configuration

Switch-Configurable Start Address

Each card has a small DIP module, that contains eight switches. The switches are used to set the card start address as follows:

Number	32K and 64K Options	64/256K Options
1	ON: A19=0; OFF: A19=1	ON: A19=0; OFF: A19=1
2	ON: A18=0; OFF: A18=1	ON: A18=0; OFF: A18=1
3	ON: A17=0; OFF: A17=1	ON: A17=0; OFF: A17=1
4	ON: A16=0: OFF: A16=1	ON: A16=0; OFF: A16=1
5	ON: A15=0; OFF: A15=1*	ON: Select 64K
6	Not used	ON: Select 128K
7	Not used	ON: Select 192K
8	Used only in 64K RAM Card*	ON: Select 256K

*Switch 8 may be set on the 64K memory expansion option to use only half the memory on the card (that is, 32K). If switch 8 is on, all 64K is accessible. If switch 8 is off, address bit A15 (as set by switch 5) is used to determine which 32K are accessible, and the 64K option behaves as a 32K option.

DIP Module Start Address

Memory Option Switch Settings

Switch settings for all memory expansion options are located in "Appendix G: Switch Settings."

The following method can be used to determine the switch settings for the 32K memory expansion option.

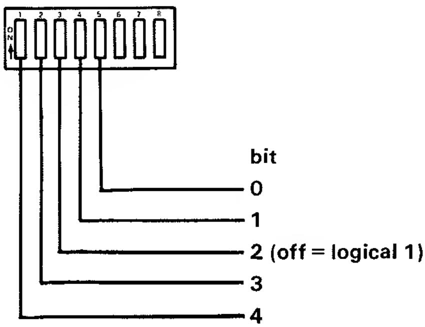
Starting Address = xxxK

$32K\overline{)xxxK}$ =Decimal value

Convert decimal value to binary

Bit.	4	3	2	1	0
Bit value . . .	16	8	4	2	1

Switch



The following method can be used to determine the switch settings for the 64K memory expansion option.

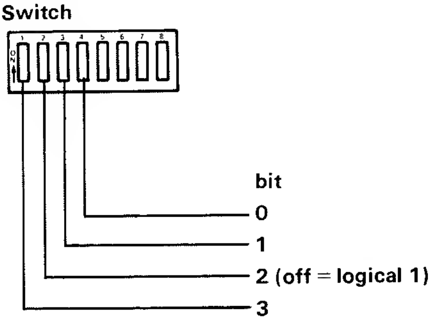
Starting Address = xxxK

$64K\overline{)xxxK}$ =Decimal value

Convert decimal value to binary

Bit.	3	2	1	0
Bit value . . .	8	4	2	1

Switch



The following method can be used to determine the switch settings for the 64/256K memory expansion option.

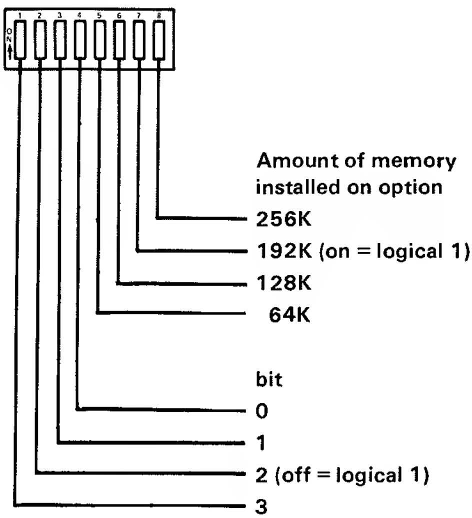
Starting Address = xxxK

$$\frac{\text{xxxK}}{\text{64K}} = \text{Decimal value}$$

Convert decimal value to binary

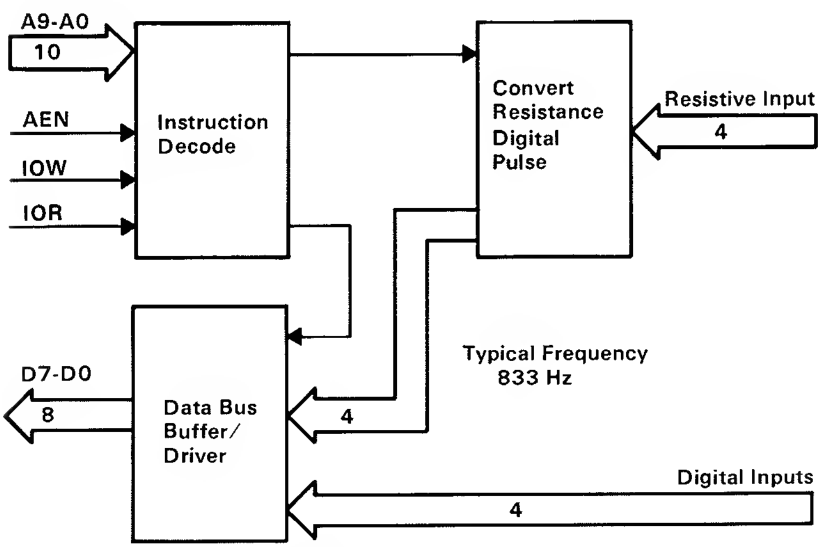
Bit.	3	2	1	0
Bit value. . .	8	4	2	1

Switch



IBM Game Control Adapter

The game control adapter allows up to four paddles or two joy sticks to be attached to the system. This card fits into one of the system board's or expansion board's expansion slots. The game control interface cable attaches to the rear of the adapter. In addition, four inputs for switches are provided. Paddle and joy stick positions are determined by changing resistive values sent to the adapter. The adapter plus system software converts the present resistive value to a relative paddle or joy stick position. On receipt of an output signal, four timing circuits are started. By determining the time required for the circuit to time-out (a function of the resistance), the paddle position can be determined. This adapter could be used as a general purpose I/O card with four analog (resistive) inputs plus four digital input points.



Game Control Adapter Block Diagram

Functional Description

Address Decode

The select on the game control adapter is generated by two 74LS138s as an address decoder. AEN must be inactive while the address is hex 201 in order to generate the select. The select allows a write to fire the one-shots or a read to give the values of the trigger buttons and one-shot outputs.

Data Bus Buffer/Driver

The data bus is buffered by a 74LS244 buffer/driver. For an In from address hex 201, the game control adapter will drive the data bus; at all other times, the buffer is left in the high impedance state.

Trigger Buttons

The trigger button inputs are read by an In from address hex 201. A trigger button is on each joy stick or paddle. These values are seen on data bits 7 through 4. These buttons default to an open state and are read as "1." When a button is pressed, it is read as "0." Software should be aware that these buttons are not debounced in hardware.

Joy Stick Positions

The joy stick position is indicated by a potentiometer for each coordinate. Each potentiometer has a range from 0 to 100 k-ohms that varies the time constant for each of the four one-shots. As this time constant is set at different values, the output of the one-shot will be of varying durations.

All four one-shots are fired at once by an Out to address hex 201. All four one-shot outputs will go true after the fire pulse and will remain high for varying times depending on where each potentiometer is set.

These four one-shot outputs are read by an In from address hex 201 and are seen on data bits 3 through 0.

I/O Channel Description

A9-A0:	Address lines 9 through 0 are used to address the game control adapter.
D7-D0:	Data lines 7 through 0 are the data bus.
IOR, IOW:	I/O read and I/O write are used when reading from or writing to an adapter (In, Out).
AEN:	When active, the adapter must be inactive and the data bus driver inactive.
+5 Vdc:	Power for the game control adapter.
GND:	Common ground.
A19-A10:	Unused.
MEMR, MEMW:	Unused.
DACK0-DACK3:	Unused.
IRQ7-IRQ2:	Unused.
DRQ3-DRQ1:	Unused.
ALE, T/C:	Unused.
CLK, OSC:	Unused.
I/O CH CK:	Unused.
I/O CH RDY:	Unused.
RESET DRV:	Unused.
−5 Vdc, +12 Vdc, −12 Vdc:	Unused.

Interface Description

The game control adapter has eight input lines, four of which are digital inputs and 4 of which are resistive inputs. The inputs are read with one In from address hex 201.

The four digital inputs each have a 1 k-ohm pullup resistor +5 Vdc. With no drives on these inputs, a 1 is read. For a 0 reading, the inputs must be pulled to ground.

The four resistive pullups, measured to +5 Vdc, will be converted to a digital pulse with a duration proportional to the resistive load, according to the following equation:

$$\text{Time} = 24.2\ \mu\text{sec} + 0.011\ (\text{r})\ \mu\text{sec}$$

The user must first begin the conversation by an Out to address hex 201. An In from address hex 201 will show the digital pulse go high and remain high for the duration according to the resistance value. All four bits (bit 3-bit 0) function in the same manner; their digital pulse will all go high simultaneously and will reset independently according to the input resistance value.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Digital Inputs				Resistive Inputs			

The typical input to the game control adapter is a set of joy sticks or game paddles.

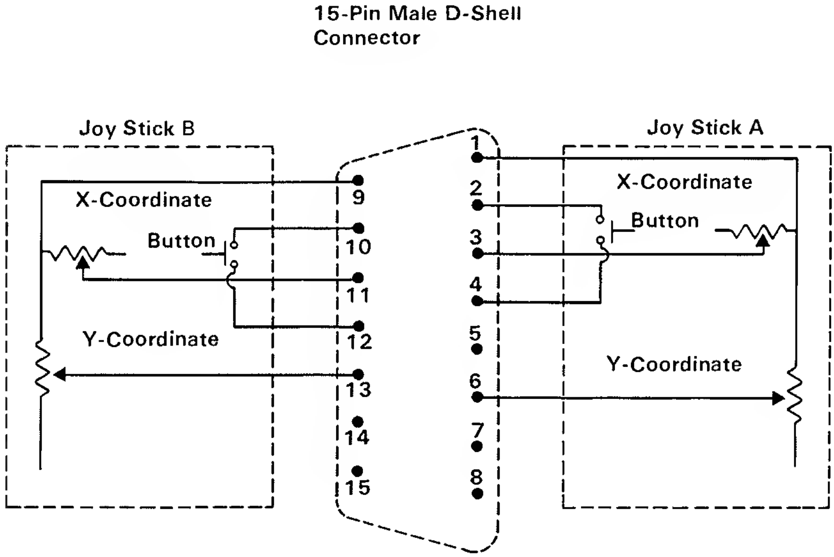
The joy sticks will typically be a set of two (A and B). These will have one or two buttons each with two variable resistances each, with a range from 0 to 100 k-ohms. One variable resistance will indicate the X-coordinate and the other variable resistance will indicate the Y-coordinate. This should be attached to give the following input data:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
B-#2 Button	B-#1 Button	A-#2 Button	A-#1 Button	B-Y Coordinate	B-X Coordinate	A-Y Coordinate	A-X Coordinate

The game paddles will have a set of two (A and B) or four (A, B, C, and D) paddles. These will have one button each and one variable resistance each, with a range of 0 to 100 k-ohms. This should be attached to give the following input data:

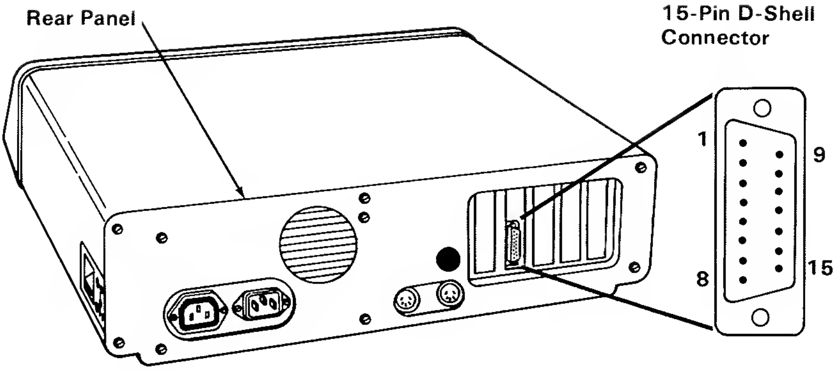
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
D Button	C Button	B Button	A Button	D Coordinate	C Coordinate	B Coordinate	A Coordinate

Refer to "Joy Stick Schematic Diagram" for attaching game controllers.



Note: Potentiometer for X- and Y-Coordinates has a range of 0 to 100 k-ohms. Button is normally open; closed when pressed.

Joy Stick Schematic Diagram



At Standard TTL Levels

External Devices → Game Control Adapter

Voltage	Adapter Pin No.
+5 Vdc	1
Button 4	2
Position 0	3
Ground	4
Ground	5
Position 1	6
Button 5	7
+5 Vdc	8
+5 Vdc	9
Button 6	10
Position 2	11
Ground	12
Position 3	13
Button 7	14
+5 Vdc	15

Connector Specifications

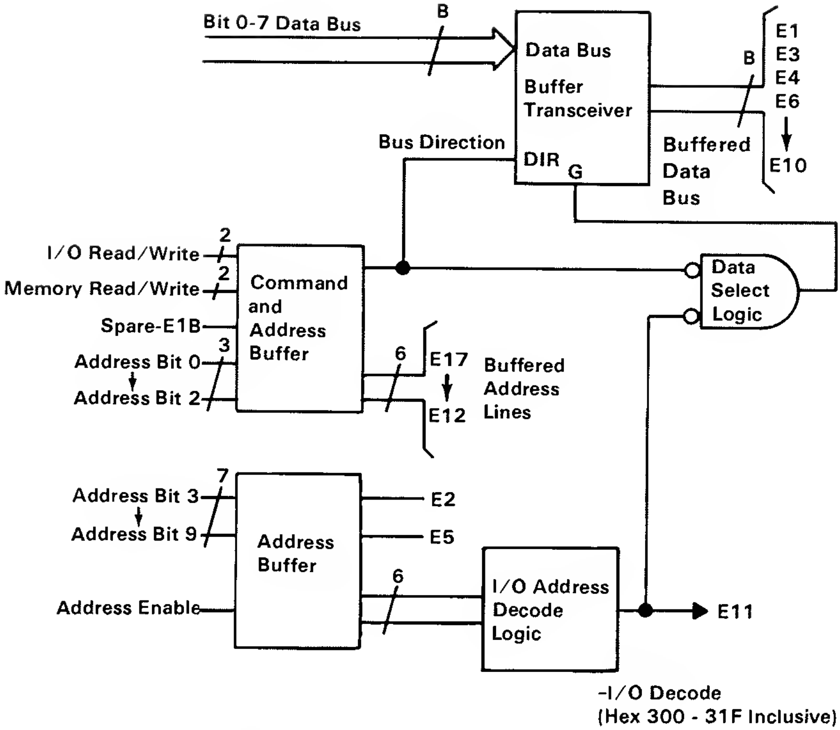
IBM Prototype Card

The prototype card is 4.2 inches (106.7 millimeters) high by 13.2 inches (335.3 millimeters) long and plugs into an expansion unit or system unit expansion slot. All system control signals and voltage requirements are provided through a 2 by 31 position card-edge tab.

The card contains a voltage bus (+5 Vdc) and a ground bus (0 Vdc). Each bus borders the card, with the voltage bus on the back (pin side) and the ground bus on the front (component side). A system interface design is also provided on the prototype card.

The prototype card can also accommodate a D-shell connector if it is needed. The connector size can range from a 9 to a 37 position connector.

Note: Install all components on the component side of the prototype card. The total width of the card including components should not exceed 0.500 inch (12.7 millimeters). If these specifications are not met, components on the prototype card may touch other cards plugged into adjacent slots.



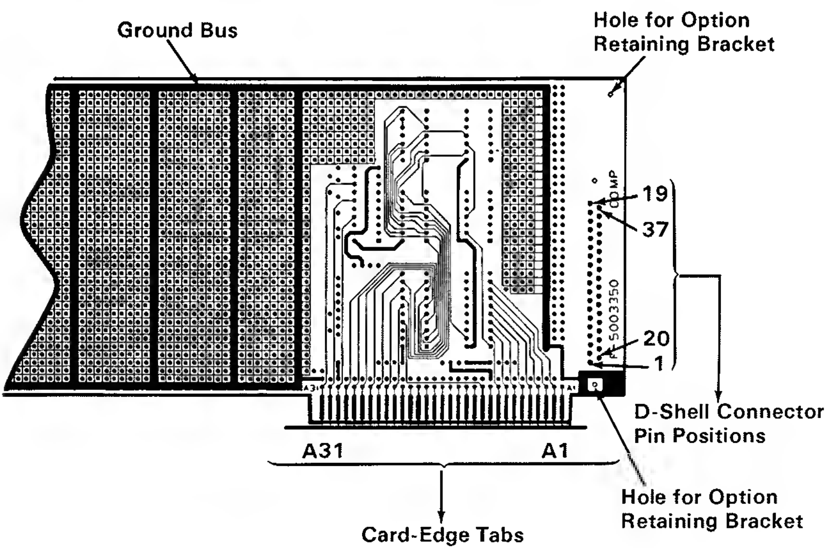
Prototype Card Block Diagram

I/O Channel Interface

The prototype card has two layers screened onto it (one on the front and one on the back). It also has 3,909 plated through-holes that are 0.040 inch (10.1 millimeters) in size and have a 0.060 inch (1.52 millimeters) pad, which is located on a 0.10 inch (2.54 millimeters) grid. There are 37 plated through-holes that are 0.048 inch (1.22 millimeters) in size. These holes are located at the rear of the card (viewed as if installed in the machine). These 37 holes are used for a 9 to 37 position D-shell connector. The card also has 5 holes that are 0.125 inch (3.18 millimeters) in size. One hole is located just above the two rows of D-shell connector holes, and the other four are located in the corners of the board (one in each corner).

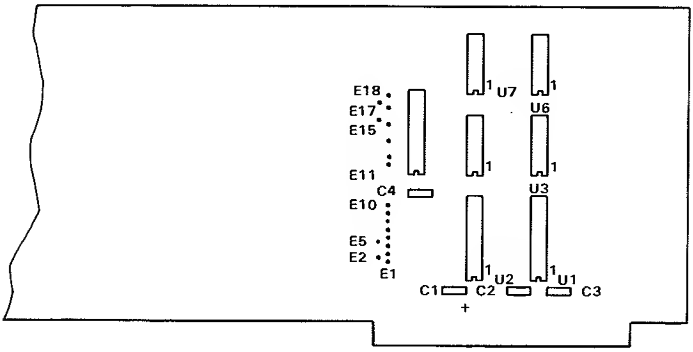
Prototype Card Layout

The component side has the ground bus [0.05 inch (1.27 millimeters) wide] screened on it and card-edge tabs that are labeled A1 through A31.



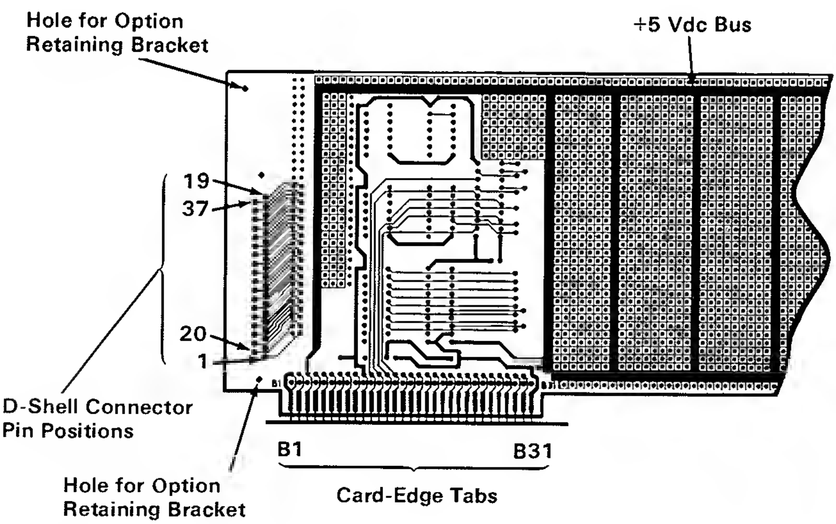
Component Side

The component side also has a silk screen printed on it that is used as a component guide for the I/O interface.



Component Side

The pin side has a +5 Vdc bus [0.05 inch (1.27 millimeters) wide] screened onto it and card-edge tabs that are labeled B1 through B31.



Pin Side

Each card-edged tab is connected to a plated through-hole by a 0.012-inch (0.3-millimeter) land. There are three ground tabs connected to the ground bus by three 0.012-inch (0.3-millimeter) lands. Also, there are two +5 Vdc tabs connected to the voltage bus by two 0.012-inch (0.3-millimeter) lands.

For additional interfacing information, refer to "I/O Channel Description" and "I/O Channel Diagram" in this manual. Also, the "Prototype Card Interface Logic Diagram" is in Appendix D of this manual. If the recommended interface logic is used, the list of TTL type numbers listed below will help you select the necessary components.

Component	TTL Number	Description
U1	74LS245	Octal Bus Transceiver
U2, U5	74LS244	Octal Buffers Line Driver/Line Receivers
U4	74LS04	Hex Inverters
U3	74LS08	Quadruple 2 - Input Positive - AND Gate
U6	74LS02	Quadruple 2 - Input Positive - NOR Gate
U7	74LS21	Dual 4 - Input Positive - AND Gate
C1		10.0 μF Tantalum Capacitor
C2, C3, C4		0.047 μF Ceramic Capacitor

System Loading and Power Limitations

Because of the number of options that may be installed in the system, the I/O bus loading should be limited to one Schottky TTL load. If the interface circuitry on the card is used, then this requirement is met.

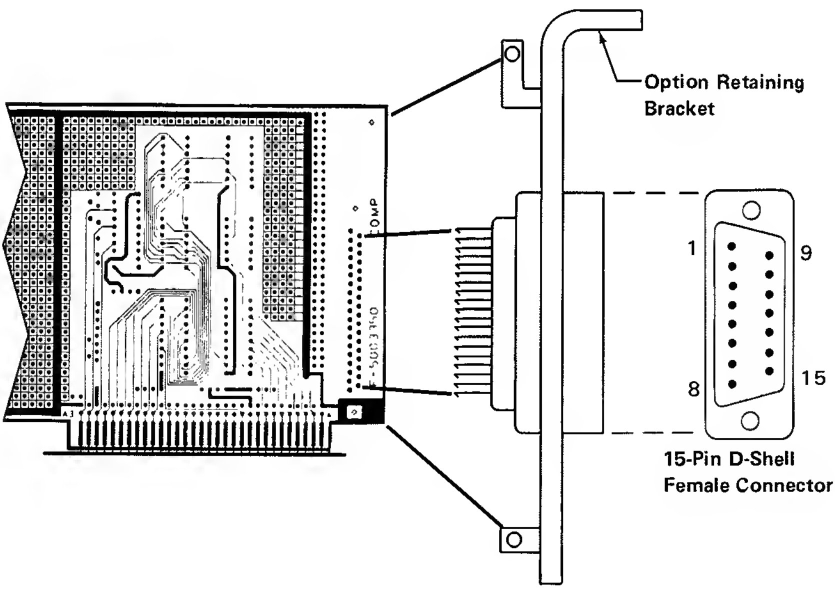
Refer to the power supply information in this manual for the power limitations to be observed.

Prototype Card External Interface

If a connector is required for the card function, then you should purchase one of the recommended connectors (manufactured by Amp) or equivalent listed below:

Connector Size	Part Number (Amp)
9-pin D-shell (Male)	205865-1
9-pin D-shell (Female)	205866-1
15-pin D-shell (Male)	205867-1
15-pin D-shell (Female)	205868-1
25-pin D-shell (Male	205857-1
25-pin D-shell (Female)	205858-1
37-pin D-shell (Male)	205859-1
37-pin D-shell (Female)	205860-1

The following example shows a 15-pin, D-shell, female connector attached to a prototype card.



Component Side

IBM Asynchronous Communications Adapter

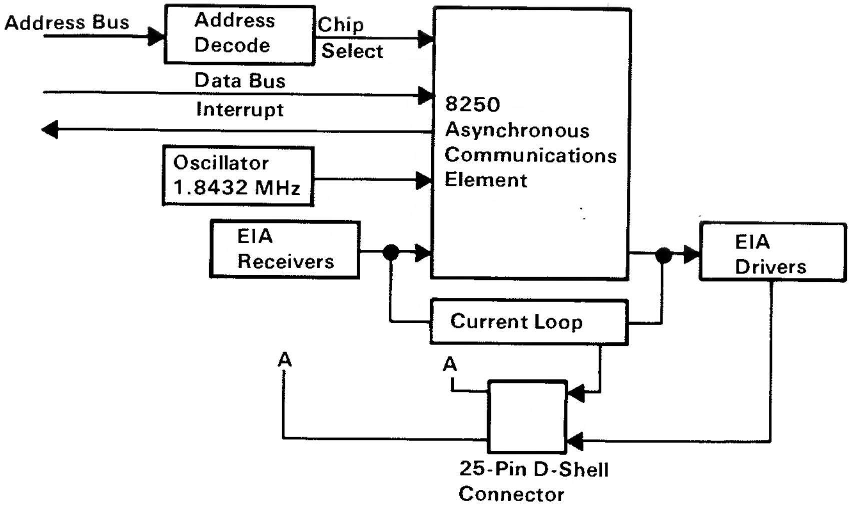
The asynchronous communications adapter system control signals and voltage requirements are provided through a 2 by 31 position card-edge tab. Two jumper modules are provided on the adapter. One jumper module selects either RS-232C or current-loop operation. The other jumper module selects one of two addresses for the adapter, so two adapters may be used in one system.

The adapter is fully programmable and supports asynchronous communications only. It will add and remove start bits, stop bits, and parity bits. A programmable baud rate generator allows operation from 50 baud to 9600 baud. Five, six, seven or eight bit characters with 1, 1-1/2, or 2 stop bits are supported. A fully prioritized interrupt system controls transmit, receive, error, line status and data set interrupts. Diagnostic capabilities provide loopback functions of transmit/receive and input/output signals.

The heart of the adapter is a INS8250 LSI chip or functional equivalent. Features in addition to those listed above are:

- Full double buffering eliminates need for precise synchronization.
- Independent receiver clock input.
- Modem control functions: clear to send (CTS), request to send (RTS), data set ready (DSR), data terminal ready (DTR), ring indicator (RI), and carrier detect.
- False-start bit detection.
- Line-break generation and detection.

All communications protocol is a function of the system microcode and must be loaded before the adapter is operational. All pacing of the interface and control signal status must be handled by the system software. The following figure is a block diagram of the asynchronous communications adapter.



Asynchronous Communications Adapter Block Diagram

Modes of Operation

The different modes of operation are selected by programming the 8250 asynchronous communications element. This is done by selecting the I/O address (hex 3F8 to 3FF primary, and hex 2F8 to 2FF secondary) and writing data out to the card. Address bits A0, A1, and A2 select the different registers that define the modes of operation. Also, the divisor latch access bit (bit 7) of the line control register is used to select certain registers.

I/O Decode (in Hex)			
Primary Adapter	Alternate Adapter	Register Selected	DLAB State
3F8	2F8	TX Buffer	DLAB=0 (Write)
3F8	2F8	RX Buffer	DLAB=0 (Read)
3F8	2F8	Divisor Latch LSB	DLAB=1
3F9	2F9	Divisor Latch MSB	DLAB=1
3F9	2F9	Interrupt Enable Register	
3FA	2FA	Interrupt Identification Registers	
3FB	2FB	Line Control Register	
3FC	2FC	Modem Control Register	
3FD	2FD	Line Status Register	
3FE	2FE	Modem Status Register	

I/O Decodes

Hex Address 3F8 to 3FF and 2F8 to 2FF											
A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	DLAB	Register
1	1/0	1	1	1	1	1	x	x	x		
							0	0	0	0	Receive Buffer (read), Transmit Holding Reg. (write)
							0	0	1	0	Interrupt Enable
							0	1	0	x	Interrupt Identification
							0	1	1	x	Line Control
							1	0	0	x	Modem Control
							1	0	1	x	Line Status
							1	1	0	x	Modem Status
							1	1	1	x	None
							0	0	0	1	Divisor Latch (LSB)
							0	0	1	1	Divisor Latch (MSB)

Note: Bit 8 will be logical 1 for the adapter designated as primary or a logical 0 for the adapter designated as alternate (as defined by the address jumper module on the adapter).

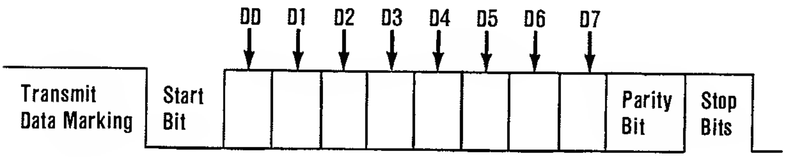
A2, A1 and A0 bits are "don't cares" and are used to select the different register of the communications chip.

Address Bits

Interrupts

One interrupt line is provided to the system. This interrupt is IRQ4 for a primary adapter or IRQ3 for an alternate adapter, and is positive active. To allow the communications card to send interrupts to the system, bit 3 of the modem control register must be set to 1 (high). At this point, any interrupts allowed by the interrupt enable register will cause an interrupt.

The data format will be as follows:



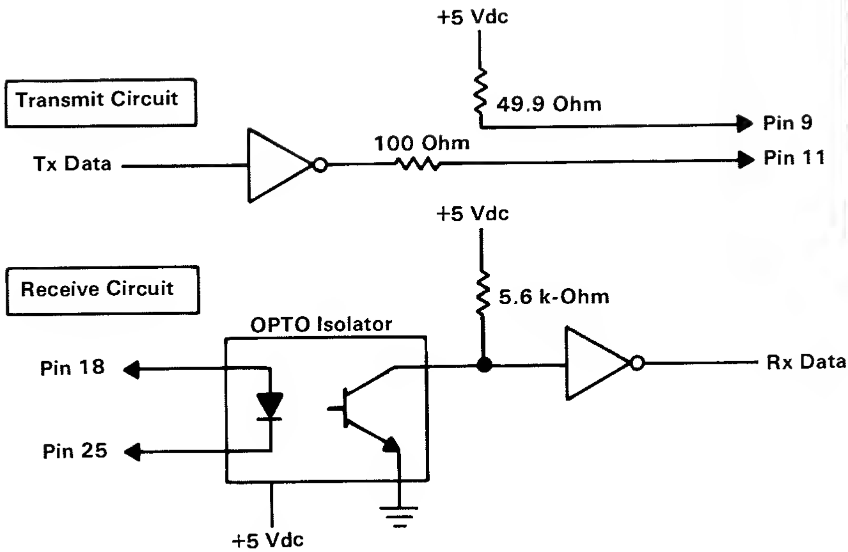
Data bit 0 is the first bit to be transmitted or received. The adapter automatically inserts the start bit, the correct parity bit if programmed to do so, and the stop bit (1, 1-1/2, or 2 depending on the command in the line-control register).

Interface Description

The communications adapter provides an EIA RS-232C-like interface. One 25-pin D-shell, male type connector is provided to attach various peripheral devices. In addition, a current loop interface is also located in this same connector. A jumper block is provided to manually select either the voltage interface, or the current loop interface.

The current loop interface is provided to attach certain printers provided by IBM that use this particular type of interface.

Pin 18 + receive current loop data
Pin 25 − receive current loop return
Pin 9 + transmit current loop return
Pin 11 − transmit current loop data



Current Loop Interface

The voltage interface is a serial interface. It supports certain data and control signals, as listed below.

Pin 2	Transmitted Data
Pin 3	Received Data
Pin 4	Request to Send
Pin 5	Clear to Send
Pin 6	Data Set Ready
Pin 7	Signal Ground
Pin 8	Carrier Detect
Pin 20	Data Terminal Ready
Pin 22	Ring Indicator

The adapter converts these signals to/from TTL levels to EIA voltage levels. These signals are sampled or generated by the communications control chip. These signals can then be sensed by the system software to determine the state of the interface or peripheral device.

Voltage Interchange Information

Interchange Voltage	Binary State	Signal Condition	Interface Control Function
Positive Voltage =	Binary (0)	= Spacing	=On
Negative Voltage =	Binary (1)	= Marking	=Off

Invalid Levels

+15 Vdc

On Function

+3 Vdc

0 Vdc Invalid Levels

-3 Vdc

Off Function

-15 Vdc

Invalid Levels

The signal will be considered in the "marking" condition when the voltage on the interchange circuit, measured at the interface point, is more negative than −3 Vdc with respect to signal ground. The signal will be considered in the "spacing" condition when the voltage is more positive than +3 Vdc with respect to signal ground. The region between +3 Vdc and −3 Vdc is defined as the transition region, and considered an invalid level. The voltage that is more negative than −15 Vdc or more positive than +15 Vdc will also be considered an invalid level.

During the transmission of data, the "marking" condition will be used to denote the binary state "1" and "spacing" condition will be used to denote the binary state "0."

For interface control circuits, the function is "on" when the voltage is more positive than +3 Vdc with respect to signal ground and is "off" when the voltage is more negative than −3 Vdc with respect to signal ground.

INS8250 Functional Pin Description

The following describes the function of all INS8250 input/output pins. Some of these descriptions reference internal circuits.

Note: In the following descriptions, a low represents a logical 0 (0 Vdc nominal) and a high represents a logical 1 (+2.4 Vdc nominal).

Input Signals

Chip Select (CS0, CS1, $\overline{\text{CS2}}$), Pins 12-14: When CS0 and CS1 are high and $\overline{\text{CS2}}$ is low, the chip is selected. Chip selection is complete when the decoded chip select signal is latched with an active (low) address strobe ($\overline{\text{ADS}}$) input. This enables communications between the INS8250 and the processor.

Data Input Strobe (DISTR, $\overline{\text{DISTR}}$) Pins 22 and 21: When DISTR is high or $\overline{\text{DISTR}}$ is low while the chip is selected, allows the processor to read status information or data from a selected register of the INS8250.

Note: Only an active DISTR or $\overline{\text{DISTR}}$ input is required to transfer data from the INS8250 during a read operation. Therefore, tie either the DISTR input permanently low or the $\overline{\text{DISTR}}$ input permanently high, if not used.

Data Output Strobe (DOSTR, $\overline{\text{DOSTR}}$), Pins 19 and 18: When DOSTR is high or $\overline{\text{DOSTR}}$ is low while the chip is selected, allows the processor to write data or control words into a selected register of the INS8250.

Note: Only an active DOSTR or $\overline{\text{DOSTR}}$ input is required to transfer data to the INS8250 during a write operation. Therefore, tie either the DOSTR input permanently low or the $\overline{\text{DOSTR}}$ input permanently high, if not used.

Address Strobe ($\overline{\text{ADS}}$), Pin 25: When low, provides latching for the register select (A0, A1, A2) and chip select (CS0, CS1, $\overline{\text{CS2}}$) signals.

Note: An active $\overline{\text{ADS}}$ input is required when the register select (A0, A1, A2) signals are not stable for the duration of a read or write operation. If not required, tie the $\overline{\text{ADS}}$ input permanently low.

Register Select (A0, A1, A2), Pins 26-28: These three inputs are used during a read or write operation to select an INS8250 register to read from or write to as indicated in the table below. Note that the state of the divisor latch access bit (DLAB), which is the most significant bit of the line control register, affects the selection of certain INS8250 registers. The DLAB must be set high by the system software to access the baud generator divisor latches.

DLAB	A2	A1	A0	Register
0	0	0	0	Receiver Buffer (Read), Transmitter Holding Register (Write)
0	0	0	1	Interrupt Enable
X	0	1	0	Interrupt Identification (Read Only)
X	0	1	1	Line Control
X	1	0	0	Modem Control
X	1	0	1	Line Status
X	1	1	0	Modem Control Status
X	1	1	1	None
1	0	0	0	Divisor Latch (Least Significant Bit)
1	0	0	1	Divisor Latch (Most Significant Bit)

Master Reset (MR), Pin 35: When high, clears all the registers (except the receiver buffer, transmitter holding, and divisor latches), and the control logic of the INS8250. Also, the state of various output signals (SOUT, INTRPT, $\overline{\text{OUT 1}}$, $\overline{\text{OUT 2}}$, $\overline{\text{RTS}}$, $\overline{\text{DTR}}$) are affected by an active MR input. Refer to the "Asynchronous Communications Reset Functions" table.

Receiver Clock (RCLK), Pin 9: This input is the 16 x baud rate clock for the receiver section of the chip.

Serial Input (SIN), Pin 10: Serial data input from the communications link (peripheral device, modem, or data set).

Clear to Send ($\overline{\text{CTS}}$), Pin 36: The $\overline{\text{CTS}}$ signal is a modem control function input whose condition can be tested by the processor by reading bit 4 (CTS) of the modem status register. Bit 0 (DCTS) of the modem status register indicates whether the $\overline{\text{CTS}}$ input has changed state since the previous reading of the modem status register.

Note: Whenever the CTS bit of the modem status register changes state, an interrupt is generated if the modem status interrupt is enabled.

Data Set Ready ($\overline{\text{DSR}}$), Pin 37: When low, indicates that the modem or data set is ready to establish the communications link and transfer data with the INS8250. The $\overline{\text{DSR}}$ signal is a modem-control function input whose condition can be tested by the processor by reading bit 5 (DSR) of the modem status register. Bit 1 (DDSR) of the modem status register indicates whether the $\overline{\text{DSR}}$ input has changed since the previous reading of the modem status register.

Note: Whenever the DSR bit of the modem status register changes state, an interrupt is generated if the modem status interrupt is enabled.

Received Line Signal Detect ($\overline{\text{RLSD}}$), Pin 38: When low, indicates that the data carrier had been detected by the modem or data set. The $\overline{\text{RLSD}}$ signal is a modem-control function input whose condition can be tested by the processor by reading bit 7 (RLSD) of the modem status register. Bit 3 (DRLSD) of the modem status register indicates whether the $\overline{\text{RLSD}}$ input has changed state since the previous reading of the modem status register.

Note: Whenever the RLSD bit of the modem status register changes state, an interrupt is generated if the modem status interrupt is enabled.

Ring Indicator ($\overline{\text{RI}}$), Pin 39: When low, indicates that a telephone ringing signal has been received by the modem or data set. The $\overline{\text{RI}}$ signal is a modem-control function input whose conditon can be tested by the processor by reading bit 6 (RI) of the modem status register. Bit 2 (TERI) of the modem status register indicates whether the $\overline{\text{RI}}$ input has changed from a low to high state since the previous reading of the modem status register.

Note: Whenever the RI bit of the modem status register changes from a high to a low state, an interrupt is generated if the modem status register interrupt is enabled.

VCC, Pin 40: +5 Vdc supply.

VSS, Pin 20: Ground (0 Vdc) reference.

Output Signals

Data Terminal Ready ($\overline{\text{DTR}}$), Pin 33: When low, informs the modem or data set that the INS8250 is ready to communicate. The DTR output signal can be set to an active low by programming bit 0 (DTR) of the modem control register to a high level. The $\overline{\text{DTR}}$ signal is set high upon a master reset operation.

Request to Send ($\overline{\text{RTS}}$), Pin 32: When low, informs the modem or data set that the INS8250 is ready to transmit data. The $\overline{\text{RTS}}$ output signal can be set to an active low by programming bit 1 (RTS) of the modem control register. The $\overline{\text{RTS}}$ signal is set high upon a master reset operation.

Output 1 ($\overline{\text{OUT 1}}$), Pin 34: User-designated output that can be set to an active low by programming bit 2 (OUT 1) of the modem control register to a high level. The $\overline{\text{OUT 1}}$ signal is set high upon a master reset operation.

Output 2 ($\overline{\text{OUT 2}}$), Pin 31: User-designated output that can be set to an active low by programming bit 3 (OUT 2) of the modem control register to a high level. The $\overline{\text{OUT 2}}$ signal is set high upon a master reset operation.

Chip Select Out (CSOUT), Pin 24: When high, indicates that the chip has been selected by active CS0, CS1, and $\overline{\text{CS2}}$ inputs. No data transfer can be initiated until the CSOUT signal is a logical 1.

Driver Disable (DDIS), Pin 23: Goes low whenever the processor is reading data from the INS8250. A high-level DDIS output can be used to disable an external transceiver (if used between the processor and INS8250 on the D7-D0 data bus) at all times, except when the processor is reading data.

Baud Out ($\overline{\text{BAUDOUT}}$), Pin 15: 16 x clock signal for the transmitter section of the INS8250. The clock rate is equal to the main reference oscillator frequency divided by the specified divisor in the baud generator divisor latches. The $\overline{\text{BAUDOUT}}$ may also be used for the receiver section by typing this output to the RCLK input of the chip.

Interrupt (INTRPT), Pin 30: Goes high whenever any one of the following interrupt types has an active high condition and is enabled through the IER: receiver error flag, received data available, transmitter holding register empty, or modem status. The INTRPT signal is reset low upon the appropriate interrupt service or a master reset operation.

Serial Output (SOUT), Pin 11: Composite serial data output to the communications link (peripheral, modem, or data set). The SOUT signal is set to the marking (logical 1) state upon a master reset operation.

Input/Output Signals

Data Bus (D7-D0), Pins 1-8: This bus comprises eight tri-state input/output lines. The bus provides bidirectional communications between the INS8250 and the processor. Data, control words, and status information are transferred through the D7-D0 data bus.

External Clock Input/Output (XTAL1, XTAL2), Pins 16 and 17: These two pins connect the main timing reference (crystal or signal clock) to the INS8250.

Programming Considerations

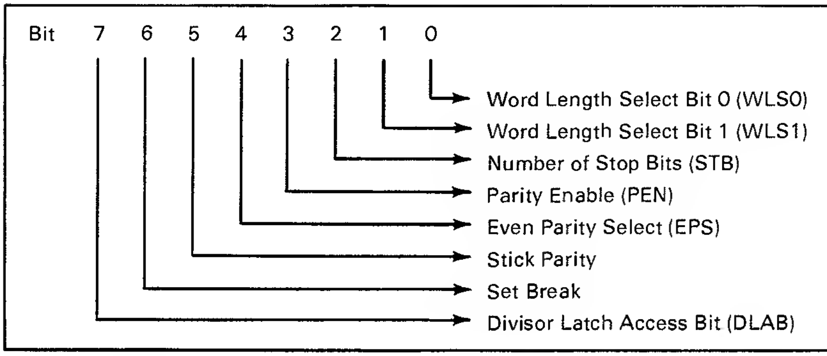
The INS8250 has a number of accessible registers. The system programmer may access or control any of the INS8250 registers through the processor. These registers are used to control INS8250 operations and to transmit and receive data. A table listing and description of the accessible registers follows.

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	Master Reset	All Bits Low (0-3 Forced and 4-7 Permanent)
Interrupt Identification Register	Master Reset	Bit 0 is High, Bits 1 and 2 Low Bits 3-7 are Permanently Low
Line Control Register	Master Reset	All Bits Low
Modem Control Register	Master Reset	All Bits Low
Line Status Register	Master Reset	Except Bits 5 and 6 are High
Modem Status Register	Master Reset	Bits 0-3 Low Bits 4-7 - Input Signal
SOUT	Master Reset	High
INTRPT (RCVR Errors)	Read LSR/MR	Low
INTRPT (RCVR Data Ready)	Read RBR/MR	Low
INTRPT (RCVR Data Ready)	Read IIR/ Write THR/MR	Low
INTRPT (Modem Status Changes)	Read MSR/MR	Low
OUT 2	Master Reset	High
RTS	Master Reset	High
DTR	Master Reset	High
OUT 1	Master Reset	High

Asynchronous Communications Reset Functions

Line-Control Register

The system programmer specifies the format of the asynchronous data communications exchange through the line-control register. In addition to controlling the format, the programmer may retrieve the contents of the line-control register for inspection. This feature simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. The contents of the line-control register are indicated and described below.



Line-Control Register (LCR)

Bits 0 and 1: These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Word Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2: This bit specifies the number of stop bits in each transmitted or received serial character. If bit 2 is a logical 0, one stop bit is generated or checked in the transmit or receive data, respectively. If bit 2 is logical 1 when a 5-bit word length is selected through bits 0 and 1, 1-1/2 stop bits are generated or checked. If bit 2 is logical 1 when either a 6-, 7-, or 8-bit word length is selected, two stop bits are generated or checked.

Bit 3: This bit is the parity enable bit. When bit 3 is a logical 1, a parity bit is generated (transmit data) or checked (receive data) between the last data word bit and stop bit of the serial data. (The parity bit is used to produce an even or odd number of 1's when the data word bits and the parity bit are summed.)

Bit 4: This bit is the even parity select bit. When bit 3 is a logical 1 and bit 4 is a logical 0, an odd number of logical 1's is transmitted or checked in the data word bits and parity bit. When bit 3 is a logical 1 and bit 4 is a logical 1, an even number of bits is transmitted or checked.

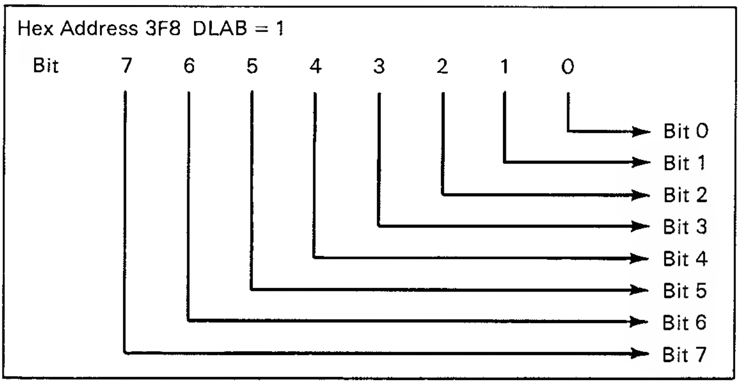
Bit 5: This bit is the stick parity bit. When bit 3 is a logical 1 and bit 5 is a logical 1, the parity bit is transmitted and then detected by the receiver as a logical 0 if bit 4 is a logical 1, or as a logical 1 if bit 4 is a logical 0.

Bit 6: This bit is the set break control bit. When bit 6 is a logical 1, the serial output (SOUT) is forced to the spacing (logical 0) state and remains there regardless of other transmitter activity. The set break is disabled by setting bit 6 to a logical 0. This feature enables the processor to alert a terminal in a computer communications system.

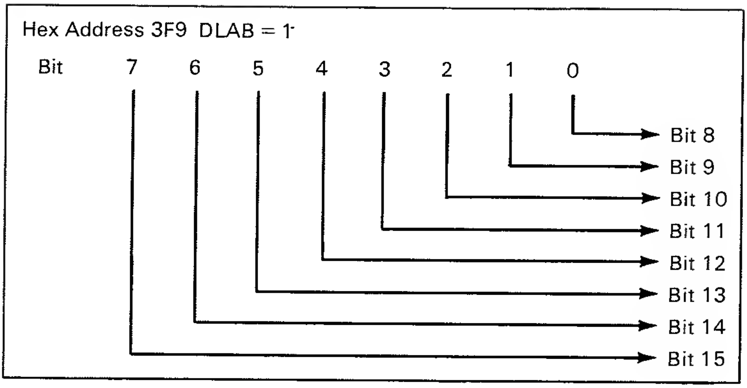
Bit 7: This bit is the divisor latch access bit (DLAB). It must be set high (logical 1) to access the divisor latches of the baud rate generator during a read or write operation. It must be set low (logical 0) to access the receiver buffer, the transmitter holding register, or the interrupt enable register.

Programmable Baud Rate Generator

The INS8250 contains a programmable baud rate generator that is capable of taking the clock input (1.8432 MHz) and dividing it by any divisor from 1 to $(2^{16}-1)$. The output frequency of the baud generator is 16 x the baud rate [divisor # = (frequency input)/(baud rate x 16)]. Two 8-bit latches store the divisor in a 16-bit binary format. These divisor latches must be loaded during initialization in order to ensure desired operation of the baud rate generator. Upon loading either of the divisor latches, a 16-bit baud counter is immediately loaded. This prevents long counts on initial load.



Divisor Latch Least Significant Bit (DLL)



Divisor Latch Most Significant Bit (DLM)

The following figure illustrates the use of the baud rate generator with a frequency of 1.8432 MHz. For baud rates of 9600 and below, the error obtained is minimal.

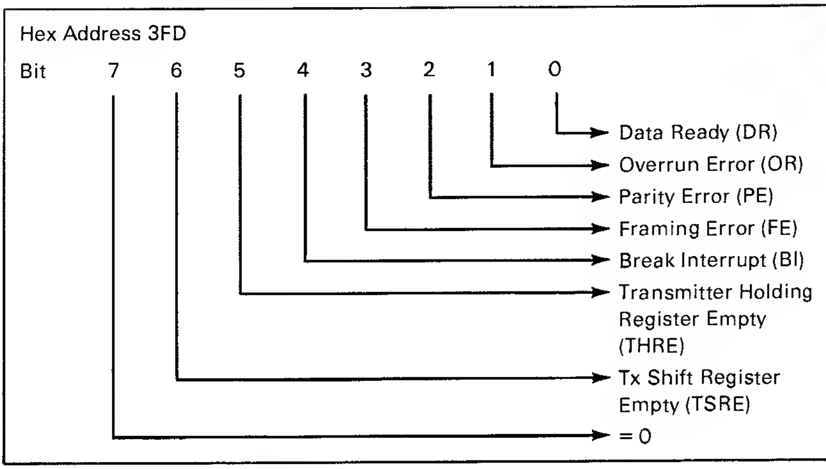
Note: The maximum operating frequency of the baud generator is 3.1 MHz. In no case should the data rate be greater than 9600 baud.

Desired Baud Rate	Divisor Used to Generate 16x Clock (Decimal)	(Hex)	Percent Error Difference Between Desired and Actual
50	2304	900	—
75	1536	600	—
110	1047	417	0.026
134.5	857	359	0.058
150	768	300	—
300	384	180	—
600	192	0C0	—
1200	96	060	—
1800	64	040	—
2000	58	03A	0.69
2400	48	030	—
3600	32	020	—
4800	24	018	—
7200	16	010	—
9600	12	00C	—

Baud Rate at 1.843 MHz

Line Status Register

This 8-bit register provides status information on the processor concerning the data transfer. The contents of the line status register are indicated and described below:



Line Status Register (LSR)

Bit 0: This bit is the receiver data ready (DR) indicator. Bit 0 is set to a logical 1 whenever a complete incoming character has been received and transferred into the receiver buffer register. Bit 0 may be reset to a logical 0 either by the processor reading the data in the receiver buffer register or by writing a logical 0 into it from the processor.

Bit 1: This bit is the overrun error (OE) indicator. Bit 1 indicates that data in the receiver buffer register was not read by the processor before the next character was transferred into the receiver buffer register, thereby destroying the previous character. The OE indicator is reset whenever the processor reads the contents of the line status register.

Bit 2: This bit is the parity error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even parity-select bit. The PE bit is set to a logical 1 upon detection of a parity error and is reset to a logical 0 whenever the processor reads the contents of the line status register.

Bit 3: This bit is the framing error (FE) indicator. Bit 3 indicates that the received character did not have a valid stop bit. Bit 3 is set to a logical 1 whenever the stop bit following the last data bit or parity is detected as a zero bit (spacing level).

Bit 4: This bit is the break interrupt (BI) indicator. Bit 4 is set to a logical 1 whenever the received data input is held in the spacing (logical 0) state for longer than a full word transmission time (that is, the total time of start bit + data bits + parity +stop bits).

Note: Bits 1 through 4 are the error conditions that produce a receiver line status interrupt whenever any of the corresponding conditions are detected.

Bit 5: This bit is the transmitter holding register empty (THRE) indicator. Bit 5 indicates that the INS8250 is ready to accept a new character for transmission. In addition, this bit causes the INS8250 to issue an interrupt to the processor when the transmit holding register empty interrupt enable is set high. The THRE bit is set to a logical 1 when a character is transferred from the transmitter holding register into the transmitter shift register. The bit is reset to logical 0 concurrently with the loading of the transmitter holding register by the processor.

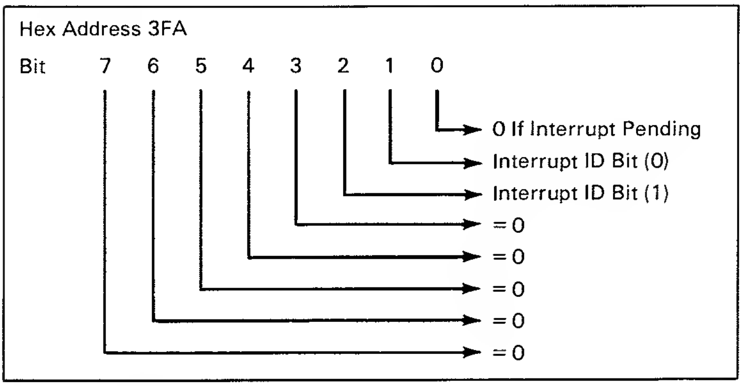
Bit 6: This bit is the transmitter shift register empty (TSRE) indicator. Bit 6 is set to a logical 1 whenever the transmitter shift register is idle. It is reset to logical 0 upon a data transfer from the transmitter holding register to the transmitter shift register. Bit 6 is a read-only bit.

Bit 7: This bit is permanently set to logical 0.

Interrupt Identification Register

The INS8250 has an on-chip interrupt capability that allows for complete flexibility in interfacing to all the popular microprocessors presently available. In order to provide minimum software overhead during data character transfers, the INS8250 prioritizes interrupts into four levels: receiver line status (priority 1), received data ready (priority 2), transmitter holding register empty (priority 3), and modem status (priority 4).

Information indicating that a prioritized interrupt is pending and the type of prioritized interrupt is stored in the interrupt identification register. Refer to the "Interrupt Control Functions" table. The interrupt identification register (IIR), when addressed during chip-select time, freezes the highest priority interrupt pending, and no other interrupts are acknowledged until that particular interrupt is serviced by the processor. The contents of the IIR are indicated and described below.



Interrupt Identification Register (IIR)

Bit 0: This bit can be used in either a hard-wired prioritized or polled environment to indicate whether an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logical 1, no interrupt is pending and polling (if used) is continued.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in the "Interrupt Control Functions" table.

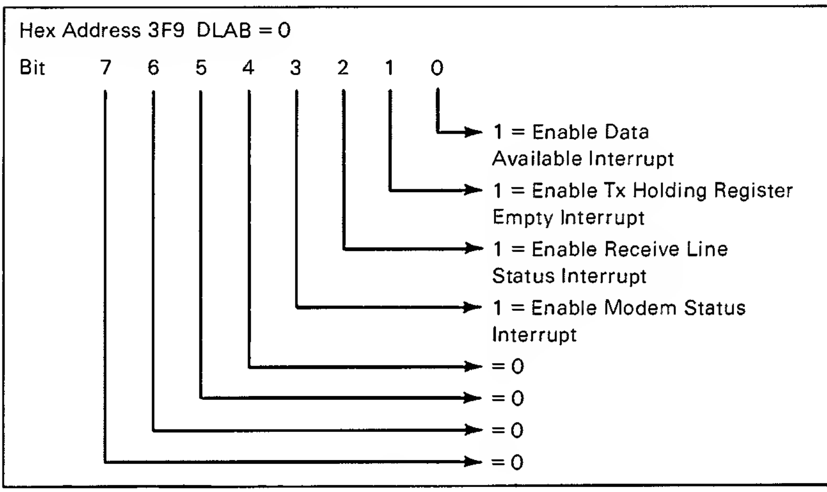
Bits 3 through 7: These five bits of the IIR are always logical 0.

Interrupt ID Register			Interrupt Set and Reset Functions			
Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	1	–	None	None	–
1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
1	0	0	Second	Received Data Available	Receiver Data Available	Reading the Receiver Buffer Register
0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register
0	0	0	Fourth	Modem Status	Clear to Send or Data Set Ready or Ring Indicator or Received Line Signal Direct	Reading the Modem Status Register

Interrupt Control Functions

Interrupt Enable Register

This eight-bit register enables the four types of interrupt of the INS8250 to separately activate the chip interrupt (INTRPT) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the interrupt enable register. Similarly, by setting the appropriate bits of this register to a logical 1, selected interrupts can be enabled. Disabling the interrupt system inhibits the interrupt identification register and the active (high) INTRPT output from the chip. All other system functions operate in their normal manner, including the setting of the line status and modem status registers. The contents of the interrupt enable register are indicated and described below:



Interrupt Enable Register (IER)

Bit 0: This bit enables the received data available interrupt when set to logical 1.

Bit 1: This bit enables the transmitter holding register empty interrupt when set to logical 1.

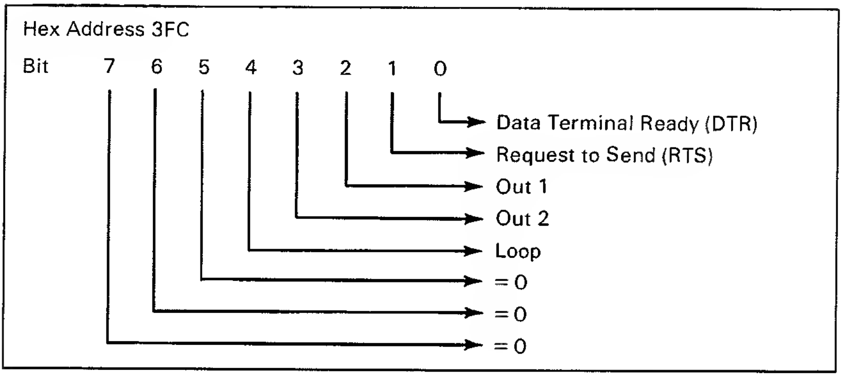
Bit 2: This bit enables the receiver line status interrupt when set to logical 1.

Bit 3: This bit enables the modem status interrupt when set to logical 1.

Bits 4 through 7: These four bits are always logical 0.

Modem Control Register

This eight-bit register controls the interface with the modem or data set (or peripheral device emulating a modem). The contents of the modem control register are indicated and described below:



Modem Control Register (MCR)

Bit 0: This bit controls the data terminal ready ($\overline{DTR}$) output. When bit 0 is set to logical 1, the $\overline{DTR}$ output is forced to a logical 0. When bit 0 is reset to a logical 0, the $\overline{DTR}$ output is forced to a logical 1.

Note: The $\overline{DTR}$ output of the INS8250 may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding modem or data set.

Bit 1: This bit controls the request to send ($\overline{RTS}$) output. Bit 1 affects the $\overline{RTS}$ output in a manner identical to that described above for bit 0.

Bit 2: This bit controls the output 1 ($\overline{\text{OUT 1}}$) signal, which is an auxiliary user-designated output. Bit 2 affects the $\overline{\text{OUT 1}}$ output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the output 2 ($\overline{\text{OUT 2}}$) signal, which is an auxiliary user-designated output. Bit 3 affects the $\overline{\text{OUT 2}}$ output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a loopback feature for diagnostic testing of the INS8250. When bit 4 is set to logical 1, the following occurs: the transmitter serial output (SOUT) is set to the marking (logical 1) state; the receiver serial input (SIN) is disconnected; the output of the transmitter shift register is "looped back" into the receiver shift register input; the four modem control inputs ($\overline{\text{CTS}}$, $\overline{\text{DRS}}$, $\overline{\text{RLSD}}$, and $\overline{\text{RI}}$) are disconnected; and the four modem control outputs ($\overline{\text{DTR}}$, $\overline{\text{RTS}}$, $\overline{\text{OUT 1}}$, and $\overline{\text{OUT 2}}$) are internally connected to the four modem control inputs. In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit- and receive-data paths of the INS8250.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. The modem control interrupts are also operational but the interrupts' sources are now the lower four bits of the modem control register instead of the four modem control inputs. The interrupts are still controlled by the interrupt enable register.

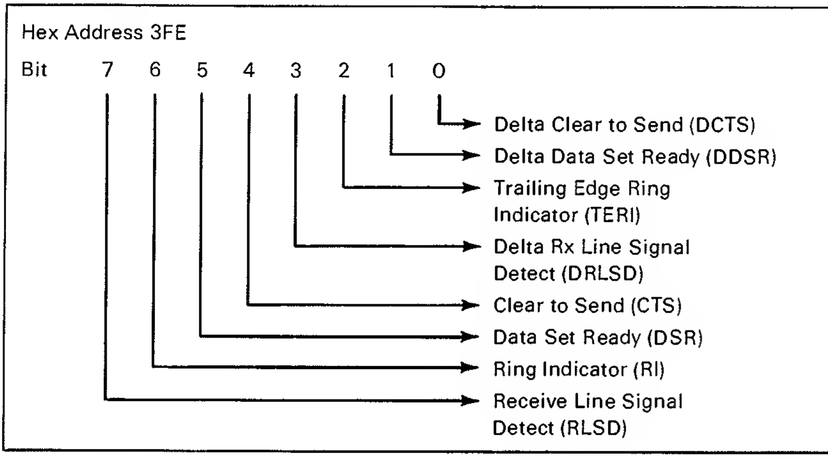
The INS8250 interrupt system can be tested by writing into the lower four bits of the modem status register. Setting any of these bits to a logical 1 generates the appropriate interrupt (if enabled). The resetting of these interrupts is the same as in normal INS8250 operation. To return to normal operation, the registers must be reprogrammed for normal operation and then bit 4 of the modem control register must be reset to logical 0.

Bits 5 through 7: These bits are permanently set to logical 0.

Modem Status Register

This eight-bit register provides the current state of the control lines from the modem (or peripheral device) to the processor. In addition to this current-state information, four bits of the modem status register provide change information. These bits are set to a logical 1 whenever a control input from the modem changes state. They are reset to logical 0 whenever the processor reads the modem status register.

The content of the modem status register are indicated and described below:



Modem Status Register (MSR)

Bit 0: This bit is the delta clear to send (DCTS) indicator. Bit 0 indicates that the $\overline{\text{CTS}}$ input to the chip has changed state since the last time it was read by the processor.

Bit 1: This bit is the delta data set ready (DDSR) indicator. Bit 1 indicates that the $\overline{\text{DRS}}$ input to the chip has changed since the last time it was read by the processor.

Bit 2: This bit is the trailing edge of ring indicator (TERI) detector. Bit 2 indicates that the $\overline{\text{RI}}$ input to the chip has changed from an on (logical 1) to an off (logical 0) condition.

Bit 3: This bit is the delta received line signal detector (DRLSD) indicator. Bit 3 indicates that the $\overline{\text{RLSD}}$ input to the chip has changed state.

Note: Whenever bit 0, 1, 2, or 3 is set to a logical 1, a modem status interrupt is generated.

Bit 4: This bit is the complement of the clear to send ($\overline{\text{CTS}}$) input. If bit 4 (LOOP) of the MCR is set to a logical 1, this is equivalent to RTS in the MCR.

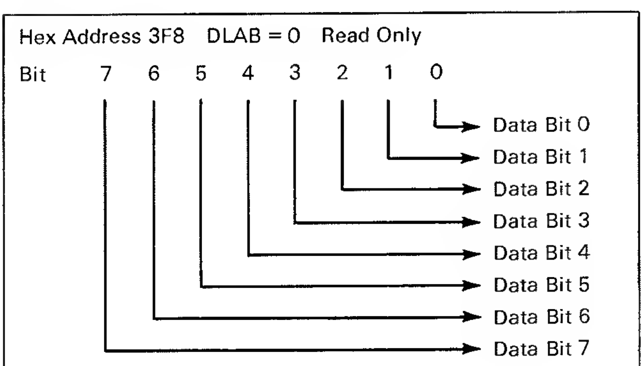
Bit 5: This bit is the complement of the data set ready ($\overline{\text{DSR}}$) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to DTR in the MCR.

Bit 6: This bit is the complement of the ring indicator ($\overline{\text{RI}}$) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to OUT 1 in the MCR.

Bit 7: This bit is the complement of the received line signal detect ($\overline{\text{RLSD}}$) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to OUT 2 of the MCR.

Receiver Buffer Register

The receiver buffer register contains the received character as defined below:

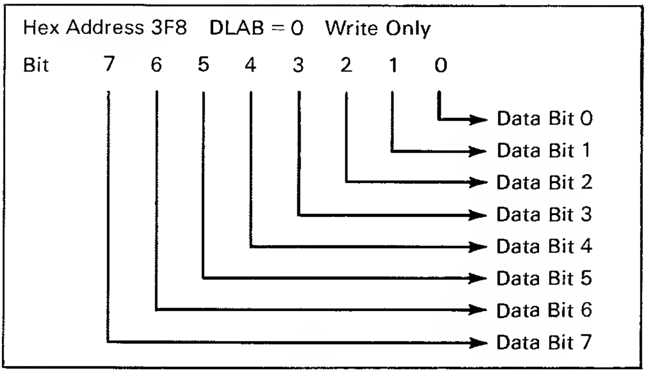


Receiver Buffer Register (RBR)

Bit 0 is the least significant bit and is the first bit serially received.

Transmitter Holding Register

The transmitter holding register contains the character to be serially transmitted and is defined below:

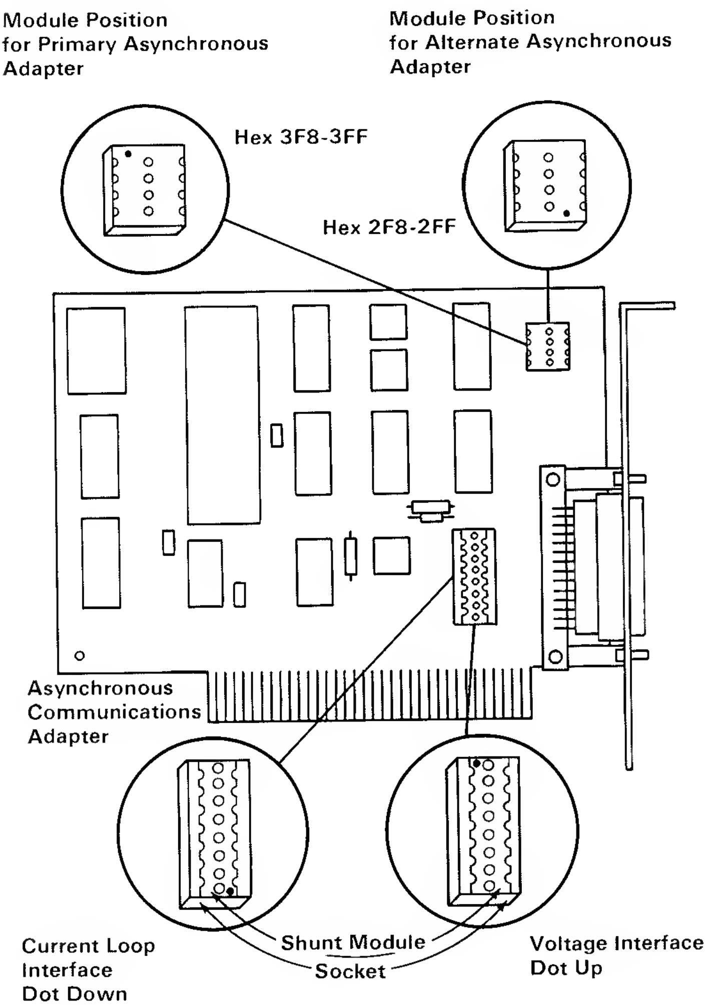


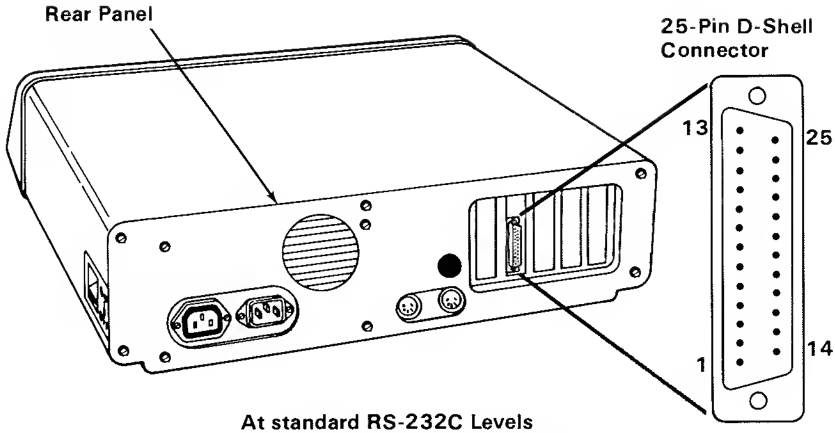
Transmitter Holding Register (THR)

Bit 0 is the least significant bit and is the first bit serially transmitted.

Selecting the Interface Format and Adapter Address

The voltage or current loop interface and adapter address are selected by plugging the programmed shunt modules with the locator dots up or down. See the figure below for the configurations.





At standard RS-232C Levels
(with exception of current loops)

External Device

Description	Pin
NC	1
Transmitted Data	2
Received Data	3
Request to Send	4
Clear to Send	5
Data Set Ready	6
Signal Ground	7
Received Line Signal Detector	8
+Transmit Current Loop Data	9
NC	10
-Transmit Current Loop Data	11
NC	12
NC	13
NC	14
NC	15
NC	16
NC	17
+Receive Current Loop Data	18
NC	19
Data Terminal Ready	20
NC	21
Ring Indicator	22
NC	23
NC	24
-Receive Current Loop Return	25

Asynchronous Communications Adapter (RS-232C)

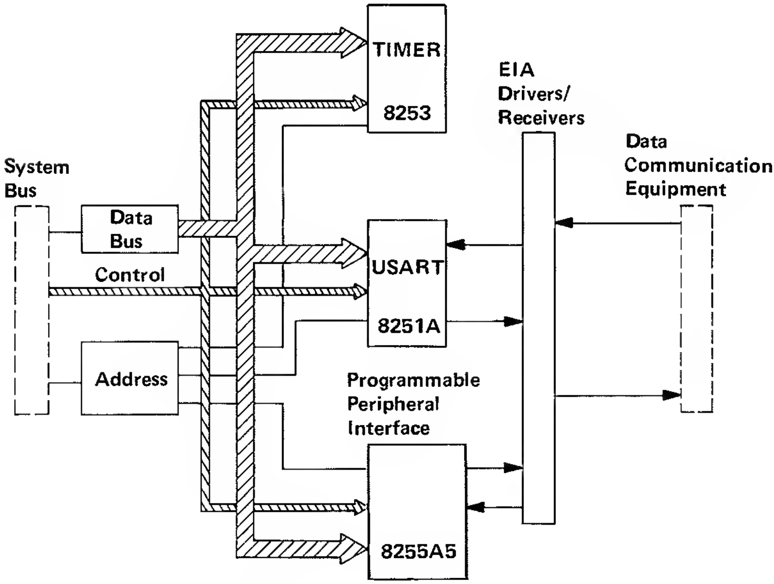
Note: To avoid inducing voltage surges on interchange circuits, signals from interchange circuits shall be used to drive inductive devices, such as relay coils.

Binary Synchronous Communications Adapter

The binary synchronous communication (BSC) adapter is a 4-inch high by 7.5-inch wide card that provides an RS232C-compatible communication interface for the IBM Personal Computer. All system control, voltage, and data signals are provided through a 2- by 31-position card-edge tab. External interface is in the form of EIA drivers and receivers connected to an RS232C, standard 25-pin, D-shell connector.

The adapter is programmed by communication software to operate in binary synchronous mode. Maximum transmission rate is 9600 bits per second (bps). The heart of the adapter is an Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART). An Intel 8255A-5 programmable peripheral interface (PPI) is also used for an expanded modem interface, and an Intel 8253-5 programmable interval timer provides time-outs and generates interrupts.

The following is a block diagram of the BSC adapter.



BSC Adapter Block Diagram

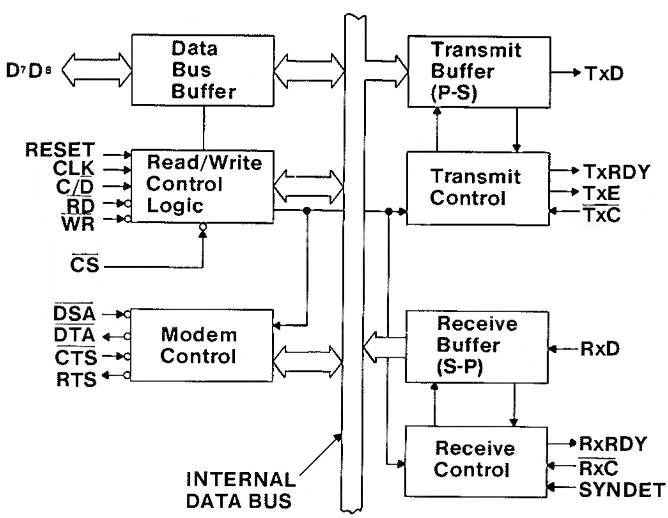
Functional Description

8251A Universal Synchronous/Asynchronous Receiver/Transmitter

The 8251A operational characteristics are programmed by the system unit's software, and it can support virtually any form of synchronous data technique currently in use. In the configuration being described, the 8251A is used for IBM's binary synchronous communications (BSC) protocol in half-duplex mode.

Operation of the 8251A is started by programming the communications format, then entering commands to tell the 8251A what operation is to be performed. In addition, the 8251A can pass device status to the system unit by doing a Status Read operation. The sequence of events to accomplish this are mode instruction, command instruction, and status read. Mode instruction must follow a master reset operation. Commands can be issued in the data block at any time during operation of the 8251A.

A block diagram of the 8251A follows:



8251A Block Diagram

Data Bus Buffer

The system unit's data bus interfaces the 8251A through the data bus buffer. Data is transferred or received by the buffer upon execution of input or output instructions from the system unit. Control words, command words, and status information are also transferred through the data bus buffer.

Read/Write Control Logic

The read/write control logic controls the transfer of information between the system unit and the 8251A. It consists of pins designated as RESET, CLK, WR, RD, C/D, and CS.

RESET: The Reset pin is gated by Port B, bit 4 of the 8255, and performs a master reset of the 8251A. The minimum reset pulse width is 6 clock cycles. Clock-cycle duration is determined by the oscillator speed of the processor.

CLK (Clock): The clock generates internal device timing. No external inputs or outputs are referenced to CLK. The input is the system board's bus clock of 4.77 MHz.

WR (Write): An input to WR informs the 8251A that the system unit is writing data or control words to it. The input is the WR signal from the system-unit bus.

RD (Read): An input to RD informs the 8251A that the processing unit is reading data or status information from it. The input is the RD signal from the system-unit bus.

C/D (Control/Data): An input on this pin, in conjunction with the WR and RD inputs, informs the 8251A that the word on the data bus is either a data character, a control word, or status information. The input is the low-order address bit from the system board's address bus.

CS (Chip Select): A low on the input selects the 8251A. No reading or writing will occur unless the device is selected. An input is decoded at the adapter from the address information on the system-unit bus.

Modem Control

The 8251A has the following input and output control signals which are used to interface the transmission equipment selected by the user.

DSR (Data Set Ready): The DSR input port is a general-purpose, 1-bit, inverting input port. The 8251A can test its condition with a Status Read operation.

CTS (Clear to Send): A low on this input enables the 8251A to transfer serial data if the TxEnable bit in the command byte is set to 1. If either a TxEnable off or CTS off condition occurs while the transmitter is in operation, the transmitter will send all the data in the USART that was written prior to the TxDisable command, before shutting down.

DTR (Data Terminal Ready): The DTR output port is a general-purpose, 1-bit, inverting output port. It can be set low by programming the appropriate bit in the command instruction word.

RTS (Request to Send): The RTS output signal is a general-purpose, 1-bit, inverting output port. It can be set low by programming the appropriate bit in the Command Instruction word.

Transmitter Buffer

The transmitter buffer accepts parallel data from the data-bus buffer, converts it to a serial bit stream, and inserts the appropriate characters or bits for the BSC protocol. The output from the transmit buffer is a composite serial stream of data on the falling edge of Transmit Clock. The transmitter will begin transferring data upon being enabled, if CTS = 0 (active). The transmit data (TxD) line will be set in the marking state upon receipt of a master reset, or when transmit enable/CTS is off and the transmitter is empty (TxEmpty).

Transmitter Control

Transmitter control manages all activities associated with the transfer of serial data. It accepts and issues the following signals, both externally and internally, to accomplish this function:

TxRDY (Transmitter Ready): This output signals the system unit that the transmitter is ready to accept a data character. The TxRDY output pin is used as an interrupt to the system unit (Level 4) and is masked by turning off Transmit Enable. TxRDY is automatically reset by the leading edge of a WR input signal when a data character is loaded from the system unit.

TxE (Transmitter Empty): This signal is used only as a status register input.

TxC (Transmit Clock): The Transmit Clock controls the rate at which the character is to be transmitted. In synchronous mode, the bit-per-second rate is equal to the TxC frequency. The falling edge of TxC shifts the serial data out of the 8251A.

Receiver Buffer

The receiver accepts serial data, converts it to parallel format, checks for bits or characters that are unique to the communication technique, and sends an "assembled" character to the system unit. Serial data input is received on the RxD (Receive Data) pin, and is clocked in on the rising edge of RxC (Receive Clock).

Receiver Control

This control manages all receiver-related activites. The parity-toggle and parity-error flip-flop circuits are used for parity-error detection, and set the corresponding status bit.

RxRDY (Receiver Ready): This output indicates that the 8251A has a character that is ready to be received by the system unit. RxRDY is connected to the interrupt structure of the system unit (Interrupt Level 3). With Receive Enable off, RxRDY is masked and held in the reset mode. To set RxRDY, the receiver must be enabled, and a character must finish assembly and be transferred to the data output register. Failure to read the received character from the RxData output register before the assembly of the next RxData character will set an overrun-condition error, and the previous character will be lost.

RxC (Receiver Clock): The receiver clock controls the rate at which the character is to be received. The bit rate is equal to the actual frequency of RxC.

SYNDET (Synchronization Detect): This pin is used for synchronization detection and may be used as either input or output, programmable through the control word. It is reset to output-mode-low upon reset. When used as an output (internal synchronization mode), the SYNDET pin will go to 1 to indicate that the 8251A has found the synchronization character in the receive mode. If the 8251A is programmed to use double synchronization characters (bisynchronization as in this application), the SYNDET pin will go to 1 in the middle of the last bit of the second synchronization character. SYNDET is automatically reset for a Status Read operation.

8255A-5 Programmable Peripheral Interface

The 8255A-5 is used on the BSC adapter to provide an expanded modem interface and for internal gating and control functions. It has three 8-bit ports, which are defined by the system during initialization of the adapter. All levels are considered plus active unless otherwise indicated. A detailed description of the ports is in "Programming Considerations" in this section.

8253-5 Programmable Interval Timer

The 8253-5 is driven by a divided-by-two system-clock signal. Its outputs are used as clocking signals and to generate inactivity timeout interrupts. These level 4 interrupts occur when either of the timers reaches its programmed terminal counts. The 8253-5 has the following outputs:

Timer 0: Not used for synchronous-mode operation.

Timer 1: Connected to port A, bit 7 of the 8255 and Interrupt Level 4.

Timer 2: Connected to port A, bit 6 of the 8255 and Interrupt Level 4.

Operation

The complete functional definition of the BSC adapter is programmed by the system software. Initialization and control words are sent out by the system to initialize the adapter and program the communications format in which it operates. Once programmed, the BSC Adapter is ready to perform its communication functions.

Transmit

In synchronous transmission, the TxD output is continuously at a mark level until the system sends its first character, which is a synchronization character to the 8251A. When the CTS line goes on, the first character is serially transmitted. All bits are shifted out on the falling edge of TxC. When the 8251A is ready to receive another character from the system for transmission, it raises TxRDY, which causes a level-4 interrupt.

Once transmission has started, the data stream at the TxD output must continue at the TxC rate. If the system does not provide the 8251A with a data character before the 8251A transmit buffers become empty, the synchronization characters will be automatically inserted in the TxD data stream. In this case, the TxE bit in the status register is raised high to signal that the 8251A is empty and that synchronization characters are being sent out. (Note that this TxE bit is in the status register, and is not the TxE pin on the 8251A). TxE does not go low when SYNC is being shifted out. The TxE status bit is internally reset by a data character being written to the 8251A.

Receive

In synchronous reception, the 8251A will achieve character synchronization, because the hardware design of the BSC adapter is intended for internal synchronization. Therefore, the SYNDET pin on the 8251A is not connected to the adapter circuits. For internal synchronization, the Enter Hunt command should be included in the first command instruction word written. Data on the RxD pin is then sampled in on the rising edge of RxC. The content of the RxD buffer is compared at every bit boundary with the first SYNC character until a match occurs. Because the 8251A has been programmed for two synchronization characters (bisynchronization), the next received character is also compared. When both SYNC characters have been detected, the 8251A ends the hunt mode and is in character synchronization. The SYNDET bit in the status register (not the SYNDET pin) is then set high, and is reset automatically by a Status Read.

Once synchronization has occurred, the 8251A begins to assemble received data bytes. When a character is assembled and ready to be transferred to memory from the 8251A, it raises RxRDY, causing an interrupt level 3 to the system.

If the system has not fetched a previous character by the time another received character is assembled (and an interrupt-level 3 issued by the adapter), the old character will be overwritten, and the overrun error flag will be raised. All error flags can be reset by an error reset operation.

Programming Considerations

Before starting data transmission or reception, the BSC adapter is programmed by the system unit to define control and gating ports, timer functions and counts, and the communication environment in which it is to operate.

Typical Programming Sequence

The 8255A-5 programmable peripheral interface (PPI) is initialized for the proper mode by selecting address hex 3A3 and writing the control word. This defines port A as an input, port B as an output for modem control and gating, and port C for 4-bit input and 4-bit output. The bit descriptions for the 8255A-5 are shown in the following figures. Using an output to port C, the adapter is then set to wrap mode, disallow interrupts, and gate external clocks (address=3A2H, data=0DH). The adapter is now isolated from the communication interface, and initialization continues.

Through bit 4 of 8255 Port B, the 8251A reset pin is brought high, held, then dropped. This resets the internal registers of the 8251A.

8255 Port A Assignments
Input Port

Address: hex 3A0 for BSC
hex 380 for Alternate BSC

Bit	Assignment
0	0 = Ring Indicate is on from Interface
1	0 = Data Carrier Defect is on from Interface
2	Oscillating = Transmit Clock Active
3	0 = Clear-to-Send is on from Interface
4	Oscillating = Receive Clock Active
5	1 = TxRDY Active
6	1 = Timer 2 Output Active
7	1 = Timer 1 Output Active

8255 Port B Assignments
Output Port

Address: hex 3A1 for BSC
hex 381 for Alternate BSC

Bit	Assignment
0	0 = Turn on Data Signal Rate Selector
1	0 = Turn on Select Standby
2	0 = Turn on Test
3	1 = Not Used
4	1 = Reset 8251A
5	1 = Gate Timer 2
6	1 = Gate Timer 1
7	1 = Gate Timers 1 and 2 to Interrupt Level 4

8255 Port C Assignments

Address: hex 3A2 for BSC
hex 382 for Alternate BSC

Bit	Assignment
0	1 = Gate Internal Clock (Output Bit)
1	1 = Gate External Clock (Output Bit)
2	1 = Electronic Wrap (Output Bit)
3	0 = Enable Timer 1 and 2, Interrupt 6 and Receive Interrupt 3
4	Oscillating = Receive Data (Input Bit)
5	Oscillating = Timer 0 Output (Input Bit)
6	0 = Test Indicate Active (Input Bit)
7	0 = BSC Adapter

The 8253-5 programmable interval timer is used in the synchronous mode to provide inactivity time-outs to interrupt the system unit after a preselected period of time has elapsed from the start of a communication operation. Counter 0 is not used for synchronous operation. Counters 1 and 2 are connected to interrupt-level 4, and are programmed to terminal-count values, which will provide the desired time delay before a level-4 interrupt is generated. These interrupts will indicate to the system software that a predetermined period of time has elapsed without a TxRDY (level 4) or RxRDY (level 3) interrupt being sent to the system unit.

The modes for each counter are programmed by selecting each timer-register address and writing the correct control word for counter operation to the adapter. The mode for counters 1 and 2 is set to 0. The terminal-count values are loaded using control-word bits D4 and D5 to select "load." The 8253-5 Control Word format is shown in the following chart.

Control Word Format Address hex 3A7

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

Definition of Control

SC – Select Counter:

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Illegal

RL – Read/Load:

RL1	RL0	
0	0	Counter Latching operation
1	0	Read/Load most significant byte only
0	1	Read/Load least significant byte only
1	1	Read/Load least significant byte first, then most significant byte

M – Mode:

M2	M1	M0		
0	0	0	Mode 0	Terminal Count Interrupt

BCD:

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

8253-5 Control Word Format

8251A Programming Procedures

After the support devices on the BSC adapter are programmed, the 8251A is loaded with a set of control words that define the communication environment. The control words are split into two formats, mode instruction, and command instruction.

Both the mode and command instructions must conform to a specified sequence for proper device operation. The mode instruction must be inserted immediately after a reset operation, before using the 8251A for data communications. The required synchronization characters for the defined communications technique are next loaded into the 8251A (usually hex 32 for BSC). All control words written to the 8251A after the mode instruction will load the command instruction. Command instructions can be written to the 8251A at any time in the data block during the operation of the 8251A. To return to the mode instruction format, the master reset bit in the command instruction word can be set to start an internal reset operation which automatically places the 8251A back into the mode instruction format. Command instructions must follow the mode instructions or synchronization characters. The following diagram is a typical data block, showing the mode instruction and command instruction.

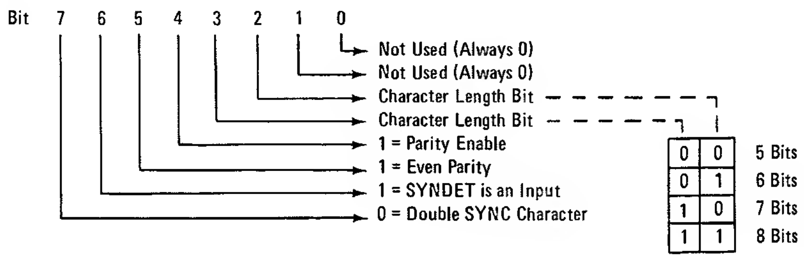
3A9 C/D = 1	Mode Instruction 1
3A9 C/D = 1	SYNC Character 1
3A9 C/$\overline{D}$ = 1	SYNC Character 2
3A9 C/$\overline{D}$ = 1	Command Instruction
3A8 C/D = 0	Data
3A9 C/D = 1	Command Instruction
3A8 C/$\overline{D}$ = 0	Data
3A9 C/D = 1	Command Instruction

Typical Data Block

Mode Instruction Definition

The mode instruction defines the general operational characteristics of the 8251A. It follows a reset operation (internal or external). Once the mode instruction has been written to the 8251A by the system unit, synchronization characters or command instructions may be written to the device. The following figure shows the format for the mode instruction.

Mode Instruction Format — **Address: Hex 3A9 for BSC; Hex 389 for Alternate BSC**

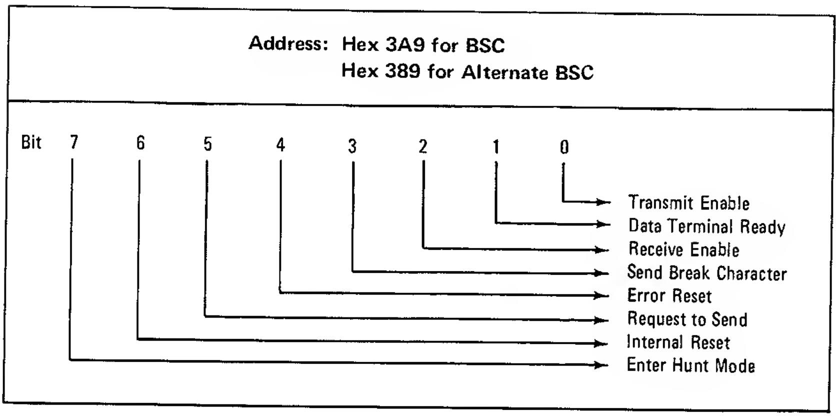


Bit 0	Not used; always = 0
Bit 1	Not used; always = 0
Bit 2 and Bit 3	These two bits are used together to define the character length. With 0 and 1 as inputs on bits 2 and 3, character lengths of 5, 6, 7, and 8 bits can be established, as shown in the preceding figure.
Bit 4	In the synchronous mode, parity is enabled from this bit. A 1 on this bit sets parity enable.
Bit 5	The parity generation/check is set from this bit. For BSC, even parity is used by having bit 5 = 1.
Bit 6	External synchronization is set by this bit. A 1 on this bit establishes synchronization detection as an input.
Bit 7	This bit establishes the mode of character synchronization. A 0 is set on this bit to give double character synchronization.

Command-Instruction Format

The command-instruction format defines a status word that is used to control the actual operation of the 8251A. Once the mode instruction has been written to the 8251A, and SYNC characters loaded, all further "Control Writes" to I/O address hex 3A9 or hex 389 will load a command instruction.

Data is transferred by accessing two I/O ports on the 8251A, ports 3A8 and 388. A byte of data can be read from port 3A8 and can be written to port 388.



Command Instruction Format

Bit 0 The Transmit Enable bit sets the function of the 8251A to either enabled (1) or disabled (0).

Bit 1 The Data Terminal Ready bit, when set to 1 will force the data terminal output to 0. This is a one-bit inverting output port.

Bit 2 The Receive Enable bit sets the function to either enable the bit (1), or to disable the bit (0).

Bit 3 The Send Break Character bit is set to 0 for normal BSC operation.

Bit 4 The Error Reset bit is set to 1 to reset error flags from the command instruction.

Bit 5 A 1 on the Request to Send bit will set the output to 0. This is a one-bit inverting output port.

Bit 6 The Internal Reset bit when set to 1 returns the 8251A to mode-instruction format.

Bit 7 The Enter Hunt bit is set to 1 for BSC to enable a search for synchronization characters.

Status Read Definition

In telecommunication systems, the status of the active device must often be checked to determine if errors or other conditions have occurred that require the processor's attention. The 8251A has a status read facility that allows the system software to read the status of the device at anytime during the functional operation. A normal read command is issued by the processor with I/O address hex 3A9 for BSC, and hex 389 for Alternate BSC to perform a status read operation.

The format for a status read word is shown in the figure below. Some of the bits in the status read format have the same meanings as external output pins so the 8251A can be used in a completely polled environment or in an interrupt-driven environment.

Address: Hex 3A9 for BSC
Hex 389 for Alternate BSC

Bits		
	0 →	TxRDY (See Note Below)
	1 →	RxRDY
	2 →	TxEmpty
	3 →	Parity Error (PE Flag On when a Parity Error Occurs)
	4 →	Overrun Error (OE Flag On when Overrun Error Occurs)
	5 →	Framing Error (Not Used for Synchronous Communications)
	6 →	SYNDET
	7 →	Data Set Ready (Indicates that DSR is at 0 Level)

Note: TxRDY status bit does not have the same meaning as the 8251A TxRDY output pin. The former is not conditioned by CTS and TxEnable. The latter is conditioned by both CTS and TxEnable.

Status Read Format

Bit 0 See the Note in the preceding chart.

Bit 1 An output on this bit means a character is ready to be received by the computers 8088 microprocessor.

Bit 2 A 1 on this bit indicates the 8251A has no characters to transmit.

Bit 3 The Parity Error bit sets a flag when errors are detected. It is reset by the error reset in the command instruction.

Bit 4 This bit sets a flag when the computers 8088 microprocessor does not read a character before another one is presented. The 8251A operation is not inhibited by this flag, but the overrun character will be lost.

Bit 5 Not used

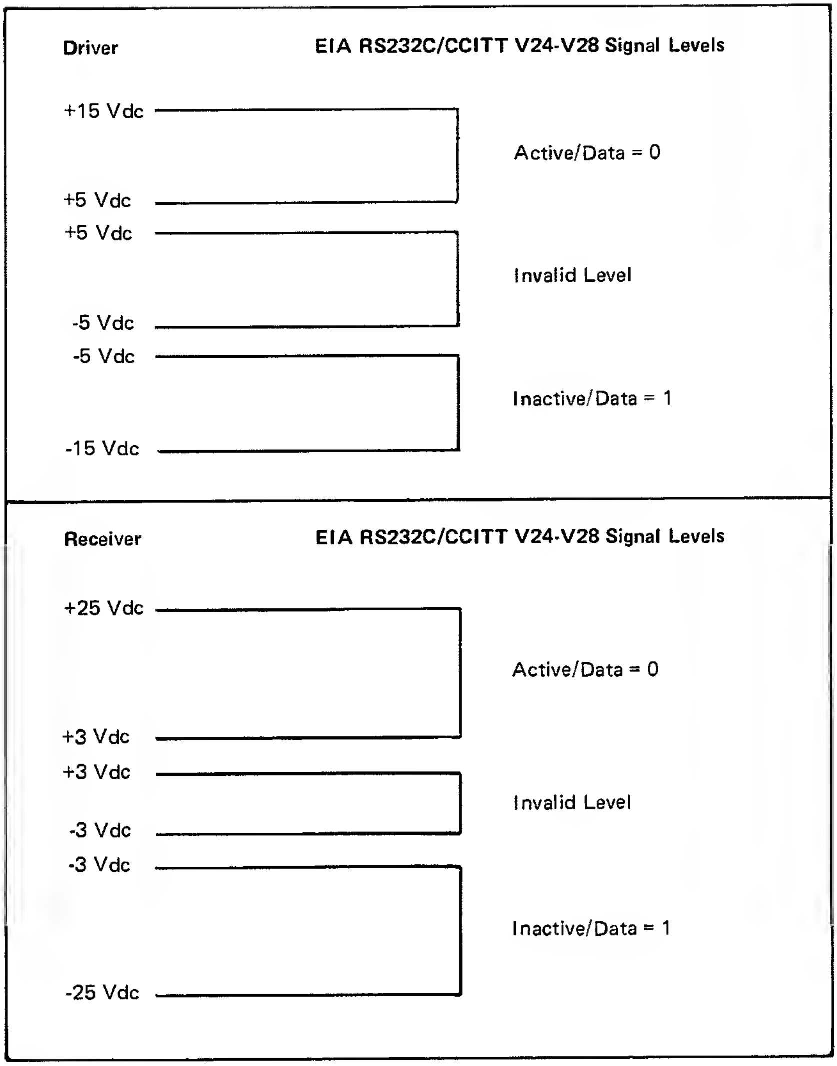
Bit 6 SYNDET goes to 1 when the synchronization character is found in receive mode. For BSC, SYNDET goes high in the middle of the last bit of the second synchronization character.

Bit 7 The Data Set Ready bit is a one bit inverting input. It is used to check modem conditions, such as data-set ready.

Interface Signal Information

The BSC adapter conforms to interface signal levels standardized by the Electronics Industry Association (EIA) RS232C Standard. These levels are shown in the following figure.

Additional lines, not standardized by the EIA, are pins 11, 18, and 25 on the interface connector. These lines are designated as Select Standby, Test, and Test Indicate. Select Standby is used to support the switched network backup facility of a modem that provides this option. Test and Test Indicate support a modem wrap function on modems that are designated for business-machine, controlled-modem wraps.



Interface Voltage Levels

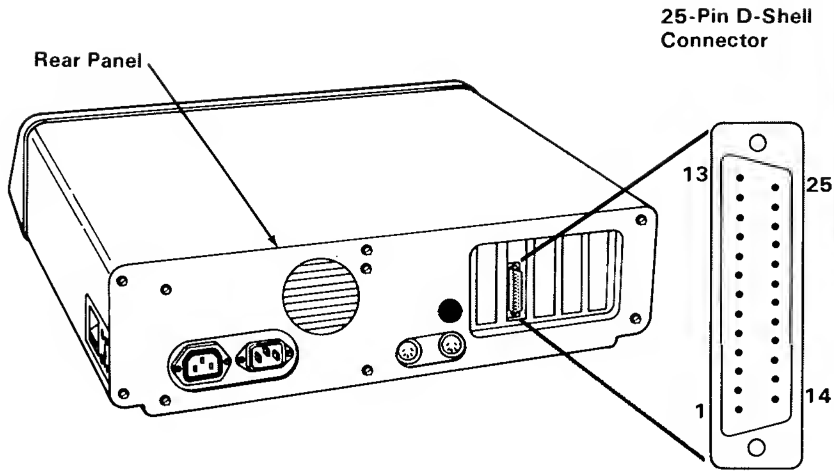
Interrupt Information

Interrupt Level 4: Transmitter Ready
Counter 1
Counter 2

Interrupt Level 3: Receiver Ready

Hex Address		Device	Register Name	Function
Primary	Alternate			
3A0	380	8255	Port A Data	Internal/External Sensing
3A1	381	8255	Port B Data	External Modem Interface
3A2	382	8255	Port C Data	Internal Control
3A3	383	8255	Mode Set	8255 Mode Initialization
3A4	384	8253	Counter 0 LSB	Not Used in Synch Mode
3A4	384	8253	Counter 0 MSB	Not Used in Synch Mode
3A5	385	8253	Counter 1 LSB	Inactivity Time-Outs
3A5	385	8253	Counter 1 MSB	Inactivity Time-Outs
3A6	386	8253	Counter 2 LSB	Inactivity Time-Outs
3A6	386	8253	Counter 2 MSB	Inactivity Time-Outs
3A7	387	8253	Mode Register	8253 Mode Set
3A8	388	8251	Data Select	Data
3A9	389	8251	Command/Status	Mode/Command USART Status

Device Address Summary



External Device — Binary Synchronous Communications Adapter

Signal Name — Description	Pin
No Connection	1
Transmitted Data	2
Received Data	3
Request to Send	4
Clear to Send	5
Data Set Ready	6
Signal Ground	7
Received Line Signal Detector	8
No Connection	9
No Connection	10
Select Standby*	11
No Connection	12
No Connection	13
No Connection	14
Transmitter Signal Element Timing	15
No Connection	16
Receiver Signal Element Timing	17
Test (IBM Modems Only)*	18
No Connection	19
Data Terminal Ready	20
No Connection	21
Ring Indicator	22
Data Signal Rate Selector	23
No Connection	24
Test Indicate (IBM Modems Only)*	25

*Not standardized by EIA (Electronics Industry Association).

Connector Specifications

Notes:

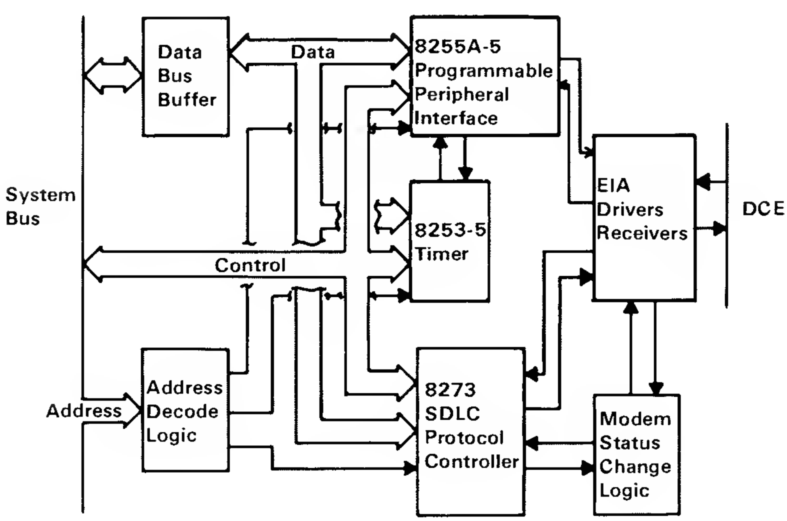
IBM Synchronous Data Link Control (SDLC) Communications Adapter

The SDLC communications adapter system control, voltage, and data signals are provided through a 2 by 31 position card edge tab. Modem interface is in the form of EIA drivers and receivers connecting to an RS232C standard 25-pin, D-shell, male connector.

The adapter is programmed by communications software to operate in a half-duplex synchronous mode. Maximum transmission rate is 9600 bits per second, as generated by the attached modem or other data communication equipment.

The SDLC adapter utilizes an Intel 8273 SDLC protocol controller and an Intel 8255A-5 programmable peripheral interface for an expanded external modem interface. An Intel 8253 programmable interval timer is also provided to generate timing and interrupt signals. Internal test loop capability is provided for diagnostic purposes.

The figure below is a block diagram of the SDLC communications adapter.



SDLC Communications Adapter Block Diagram

The 8273 SDLC protocol control module has the following key features:

- Automatic frame check sequence generation and checking.
- Automatic zero bit insertion and deletion.
- TTL compatibility.
- Dual internal processor architecture, allowing frame level command structure and control of data channel with minimal system processor intervention.

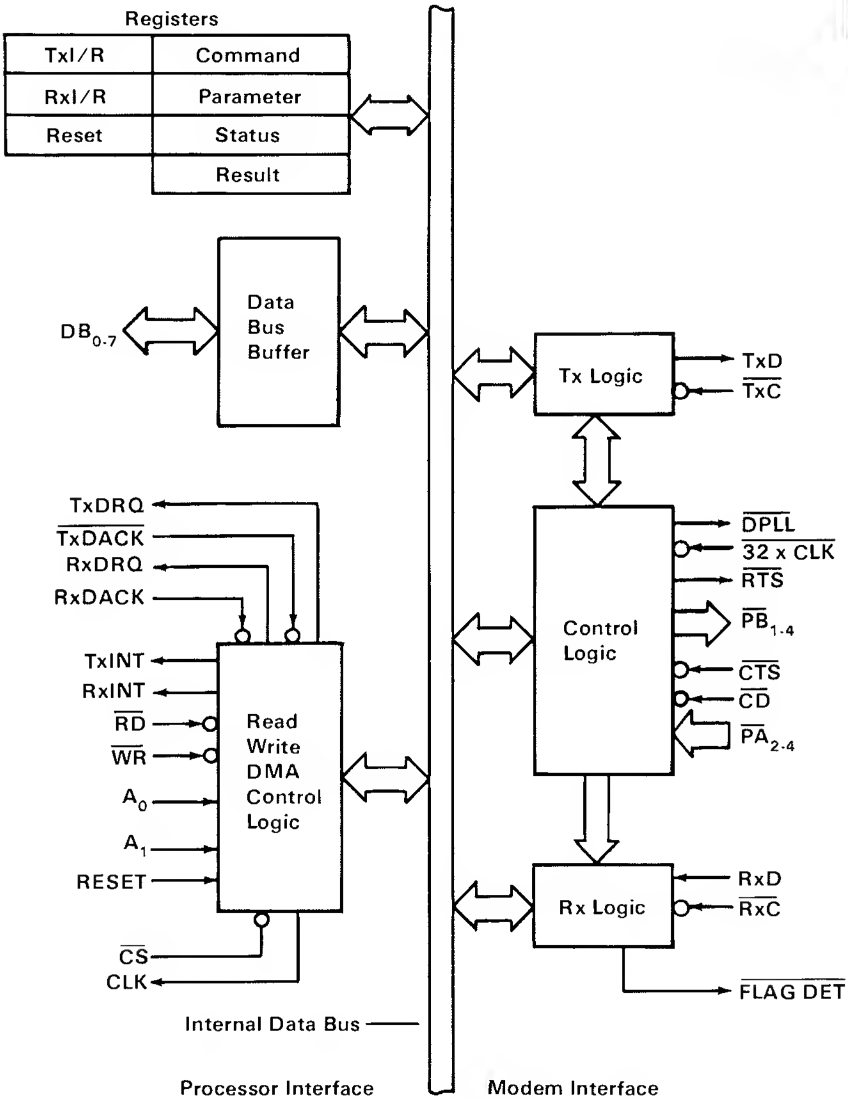
The 8273 SDLC protocol controller operations, whether transmission, reception, or port read, are each comprised of three phases:

Command	Commands and/or parameters for the required operation are issued by the processor.
Execution	Executes the command, manages the data link, and may transfer data to or from memory utilizing direct memory access (DMA), thus freezing the processor except for minimal interruptions.
Result	Returns the outcome of the command by returning interrupt results.

Support of the controller operational phases is through internal registers and control blocks of the 8273 controller.

8273 Protocol Controller Structure

The 8273 module consists of two major interfaces: the processor interface and the modem interface. A block diagram of the 8273 protocol controller module follows.



8273 SDLC Protocol Control Block Diagram

Processor Interface

The processor interface consists of four major blocks: the control/read/write logic (C/R/W), internal registers, data transfer logic, and data bus buffers.

Control/Read/Write Logic

The control/read/write logic is used by the processor to issue commands to the 8273. Once the 8273 receives and executes a command, it returns the results using the C/R/W logic. The logic is supported by seven registers which are addressed by A0, A1, RD, and WR, in addition to CS. A0 and A1 are the two low-order bits of the adapter address-byte. RD and WR are the processor read and write signals present on the system control bus. CS is the chip select, also decoded by the adapter address logic. The table below shows the address of each register using the C/R/W logic.

Address Inputs		Control Inputs			Register
A0	A1	CS	WR	RD	
0	0	0	0	1	Command
0	0	0	1	0	Status
0	1	0	0	1	Parameter
0	1	0	1	0	Result
1	0	0	0	1	Reset
1	0	0	1	0	TxI/R
1	1	0	0	1	None
1	1	0	1	0	RxI/R

8273 SDLC Protocol Controller Register Selection

8273 Control/Read/Write Registers

Command	Operations are initialized by writing the appropriate command byte into this register.
Status	This register provides the general status of the 8273. The status register supplies the processor/adapter handshaking necessary during various phases of the 8273 operation.
Parameter	Additional information that is required to process the command is written into this register. Some commands require more than one parameter.
Immediate Result (Result)	Commands that execute immediately produce a result byte in this register, to be read by the processor.
Transmit Interrupt Results (TxI/R)	Results of transmit operations are passed to the processor from this register. This result generates an interrupt to the processor when the result becomes available.
Receiver Interrupt Results (Rx/I/R)	Results of receive operations are passed to the processor from this register. This result generates an interrupt to the processor when the result becomes available.
Reset	This register provides a software reset function for the 8273.

The other elements of the C/R/W logic are the interrupt lines (RxINT and TxINT). Interrupt priorities are listed in the "Interrupt Information" table in this section. These lines signal the processor that either the transmitter or the receiver requires service (results should be read from the appropriate register), or a data transfer is required. The status of each interrupt line is also reflected by a bit in the status register, so non-interrupt driven operation is also possible by the communication software examining these bits periodically.

Data Interfaces

The 8273 supports two independent data interfaces through the data transfer logic: received data and transmitted data. These interfaces are programmable for either DMA or non-DMA data transfers. Speeds below 9600 bits-per-second may or may not require DMA, depending on the task load and interrupt response time of the processor. The processor DMA controller is used for management of DMA data transfer timing and addressing. The 8273 handles the transfer requests and actual counts of data-block lengths. DMA level 1 is used to transmit and receive data transfers. Dual DMA support is not provided.

Elements of Data Transfer Interface

TxDRQ/RxDRQ	This line requests a DMA to or from memory and is asserted by the 8273.
TxDACK/RxDACK	This line notifies the 8273 that a request has been granted and provides access to data regions. This line is returned by the DMA controller (DACK1 on the system unit control bus is connected to TxDACK/RxDACK on the 8273).
RD (Read)	This line indicates data is to be read from the 8273 and placed in memory. It is controlled by the processor DMA controller.
WR (Write)	This line indicates if data is to be written to the 8273 from memory and is controlled by the processor DMA controller.

To request a DMA transfer, the 8273 raises the DMA request line. Once the DMA controller obtains control of the system bus, it notifies the 8273 that the DRQ is granted by returning DACK, and WR or RD, for a transmit or receive operation, respectively. The DACK and WR or RD signals transfer data between the 8273 and memory, independent of the 8273 chip-select pin (CS). This "hard select" of data into the transmitter or out of the receiver alleviates the need for the normal transmit and receive data registers, addressed by a combination of address lines, CS, and WR or RD.

Modem Interface

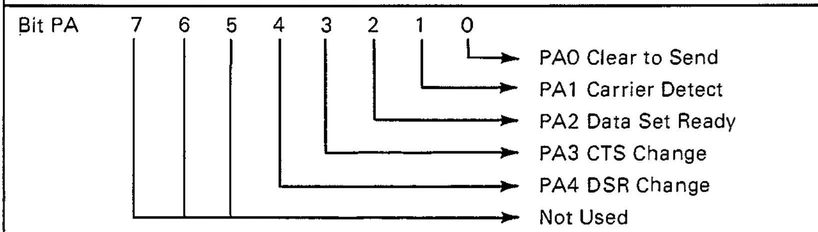
The modem interface of the 8273 consists of two major blocks: the modem control block and the serial data timing block.

Modem Control Block

The modem control block provides both dedicated and user-defined modem control function. EIA inverting drivers and receivers are used to convert TTL levels to EIA levels.

Port A is a modem control input port. Bits PA0 and PA1 have dedicated functions.

8273 Port A (Modem Control Input Port)



Bit PA0 This bit reflects the logical state of the clear to send (CTS) pin. The 8273 waits until CTS is active before it starts transmitting a frame. If CTS goes inactive while transmitting, the frame is aborted and the processor is interrupted. A CTS failure will be indicated in the appropriate interrupt-result register.

Bit PA1 This bit reflects the logical state of the carrier detect pin (CD). CD must be active in sufficient time for reception of a frame's address field. If CD is lost (goes inactive) while receiving a frame, an interrupt is generated with a CD failure result.

Bit PA2 This bit is a sense bit for data set ready (DSR).

Bit PA3 This bit is a sense bit to detect a change in CTS.

Bit PA4 — This bit is a sense bit to detect a change in data set ready.

Bits PA5 to PA7 — These bits are not used and each is read as a 1 for a read port A command.

Port B is a modem control output port. Bits PB0 and PB5 are dedicated function pins.

8273 Port B (Modem Control Output Port)

Bit PB	Function
0	PB0 - Request to Send
1	PB1 - Reserved
2	PB2 - Data Terminal Ready
3	PB3 - Reserved
4	PB4 - Reserved
5	PB5 - Flag Detect
6	PB6 - Not Used
7	PB7 - Not Used

Bit PB0 — This bit represents the logical state of request to send (RTS). This function is handled automatically by the 8273.

Bit PB1 — Reserved.

Bit PB2 — Used for data terminal ready.

Bit PB3 — Reserved.

Bit PB4 — Reserved.

Bit PB5 — This bit reflects the state of the flag detect pin. This pin is activated whenever an active receiver sees a flag character.

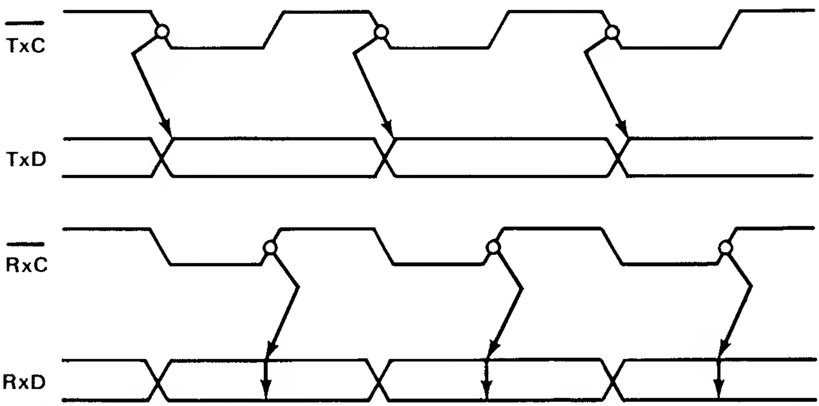
Bit PB6 — Not used.

Bit PB7 — Not used.

Serial Data Timing Block

The serial data timing block is comprised of two sections: the serial data logic and the digital phase locked loop (DPLL).

Elements of the serial data logic section are the data pins TxD (transmitted data output) and RxD (received data input), and the respective clocks. The leading edge of TxC generates new transmitted data and the trailing edge of RxC is used to capture the received data. The figure below shows the timing for these signals.



8273 SDLC Protocol Controller Transmit/Receive Timing

The digital phase locked loop provided on the 8273 controller module is utilized to capture looped data in proper synchronization during wrap operations performed by diagnostics.

8255A-5 Programmable Peripheral Interface

The 8255A-5 contains three 8-bit ports. Descriptions of each bit of these ports are as follows:

8255A-5 Port A Assignments* — **Hex Address 380**

Bit	
0	0 = Ring Indicator is on from Interface
1	0 = Data Carrier Detect is on from Interface
2	Oscillating = Transmit Clock Active
3	0 = Clear to Send is on from Interface
4	Oscillating = Receive Clock Active
5	1 = Modem Status Changed
6	1 = Timer 2 Output Active
7	1 = Timer 1 Output Active

*Port A is defined as an input port

8255A-5 Port B Assignments* — **Hex Address 381**

Bit	
0	0 = Turn On Data Signal Rate Select at Modem Interface
1	0 = Turn On Select Standby at Modem Interface
2	0 = Turn On Test
3	1 = Reset Modem Status Changed Logic
4	1 = Reset 8273
5	1 = Gate Timer 2
6	1 = Gate Timer 1
7	1 = Enable Level 4 Interrupt

*Port B is defined as an output port

8255A-5 Port C Assignments* Hex Address 382

Bit	Assignment
0	1 = Gate Internal Clock (Output Bit)
1	1 = Gate External Clock (Output Bit)
2	1 = Electronic Wrap (Output Bit)
3	0 = Gate Interrupts 3 and 4 (Output Bit)
4	Oscillating = Receive Data (Input Bit)
5	Oscillating = Timer 0 Output (Input bit)
6	0 = Test Indicate Active (Input Bit)
7	Not Used

*Port C is defined for internal control and gating functions. It has three input and four output bits. The four output bits are defined during initialization, but only three are used.

8253-5 Programmable Interval Timer

The 8253-5 is driven by a processor clock signal divided by two. It has the following output:

Timer 0 Programmed to generate a square wave signal, used as an input to timer 2. Also connected to 8253 port C, bit 5.

Timer 1 Connected to 8255 port A, bit 7, and interrupt level 4.

Timer 2 Connected to 8255 port A, bit 6, and interrupt level 4.

Programming Considerations

The software aspects of the 8273 involve the communication of both commands from the processor to the 8273 and the return of results of those commands from the 8273 to the processor. Due to the internal processor architecture of the 8273, this system unit/8273 communication is basically a form of interprocessor communication, and must be considered when programming for the SDLC communications adapter.

The protocol for this interprocessor communication is implemented through use of handshaking supplied in the 8273 status register. The bit defintions of this register are shown below.

8273 Status Register Format		Hex Address 388
Bit		
0	TxIRA	1 = TxINT Result Available
1	RxIRA	1 = RxINT Result Available
2	TxINT	1 = Tx Interrupt
3	RxINT	1 = Rx Interrupt
4	CRBF	1 = Command Result Buffer Full
5	CPBF	1 = Command Parameter Buffer Full
6	CBF	1 = Command Buffer Full
7	CBSY	1 = Command Busy

Bit 0 This bit is the transmitter interrupt result available (TxIRA) bit. This bit is set when the 8273 places an interrupt-result byte in the TxI/R register, and reset when the processor reads the TxI/R register.

Bit 1 This bit is the receiver interrupt result available (RxIRA) bit. It is the corresponding result-available bit for the receiver. It is set when the 8273 places an interrupt-result byte in the RxI/R register and reset when the processor reads the register.

Bit 2 This bit is the transmitter interrupt (TxINT) bit and reflects the state of the TxINT pin. TxINT is set by the 8273 whenever the transmitter needs servicing, and reset when the processor reads the result or performs the data transfer.

Bit 3 This bit is the receiver interrupt (RxINT) bit and is identical to the TxINT, except action is initiated based on receiver interrupt-sources.

Bit 4 This bit is the command result buffer full (CRBF) bit. It is set when the 8273 places a result from an immediate-type command in the result register, and reset when the processor reads the result or performs the data transfer.

Bit 5 This bit is the command parameter buffer full (CPBF) bit and indicates that the parameter register contains a parameter. It is set when the processor deposits a parameter in the parameter register, and reset when the 8273 accepts the parameter.

Bit 6 This bit is the command buffer full (CBF) bit and, when set, it indicates that a byte is present in the command register. This bit is normally not used.

Bit 7 This bit is the command busy (CBSY) bit and indicates when the 8273 is in the command phase. It is set when the processor writes a command into the command register, starting the command phase. It is reset when the last parameter is deposited in the parameter register and accepted by the 8273, completing the command phase.

Initializing the Adapter (Typical Sequence)

Before initialization of the 8273 protocol controller, the support devices on the card must be initialized to the proper modes of operation.

Configuration of the 8255A-5 programmable peripheral interface is accomplished by selecting the mode-set address for the 8255 (see the "SDLC Communications Adapter Device Addresses" table later in this section) and writing the appropriate control word to the device (hex 98) to set ports A, B, and C to the modes described previously in this section.

Next, a bit pattern is output to port C which disallows interrupts, sets wrap mode on, and gates the external clock pins (address = hex 382, data = hex 0D). The adapter is now isolated from the communications interface.

Using bit 4 of port B, the 8273 reset line is brought high, held and then dropped. This resets the internal registers of the 8273.

The 8253-5's counter 1 and 2 terminal-count values are now set to values which will provide the desired time delay before a level 4 interrupt is generated. These interrupts may be used to indicate to the communication software that a pre-determined period of time has elapsed without a result interrupt (interrupt level 3). The terminal count-values for these counters are set for any time delay which the programmer requires. Counter 0 is also set at this time to mode 3 (generates square wave signal, used to drive counter 2 input).

To setup the counter modes, the address for the 8253 counter mode register is selected (see the "SDLC Communications Adapter Device Addresses" table, later in this section), and the control word for each individual counter is written to the device separately. The control-word format and bit definitions for the 8253 are shown below. Note that the two most-significant bits of the control word select each individual counter, and each counter mode is defined separately.

Once the support devices have been initialized to the proper modes and the 8273 has been reset, the 8273 protocol controller is ready to be configured for the operating mode that defines the communications environment in which it will be used.

Control Word Format

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

Definitions of Control

SC - Select Counter:

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Illegal

RL - Read/Load:

RL1	RL0	
0	0	Counter Latching operation
1	0	Read/Load most significant byte (MSB)
0	1	Read/Load least significant byte (LSB)
1	1	Read/Load least significant byte first, then most significant byte.

M - Mode:

M2	M1	M0	Mode
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

BCD:

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

8253-5 Programmable Interval Timer Control Word

Initialization/Configuration Commands

The initialization/configuration commands manipulate internal registers of the 8273, which define operating modes. After chip reset, the 8273 defaults to all 1's in the mode registers. The initialization/configuration commands either set or reset specified bits in the registers depending on the type of command. One parameter is required with the commands. The parameter is actually the bit pattern (mask) used by the set or reset command to manipulate the register bits.

Set commands perform a logical OR operation of the parameter (mask) of the internal register. This mask contains 1's where register bits are to be set. Zero (0's) in the mask cause no change to the corresponding register bit.

Reset commands perform a logical AND operation of the parameter (mask) and internal register. The mask 0 is reset to register bit, and 1 to cause no change.

The following are descriptions of each bit of the operating, serial I/O, one-bit delay, and data transfer mode registers.

Operating Mode Register

8273 Operating Mode Register Format

Bit	Function
0	1 = Flag Stream Mode
1	1 = Two Preframe Sync Characters
2	1 = Buffered Mode
3	1 = Enable Early Tx Interrupt
4	1 = EOP Interrupt Enable
5	1 = HDLC Abort Enable
6	Not Used
7	Not Used

Bit 0 If bit 0 is set to a 1, flags are sent immediately if the transmitter was idle when the bit was set. If a transmit or transmit-transparent command was active, flags are sent immediately after transmit completion. This mode is ignored if loop transmit is active or the one-bit-delay mode register is set for one-bit delay. If bit 0 is reset (to 0), the transmitter sends idles on the next character boundary if idle or, after transmission is complete, if the transmitter was active at bit-0 reset time.

Bit 1 If bit 1 is set to a 1, the 8273 sends two characters before the first flag of a frame. These characters are hex 00 if NRZI is set or hex 55 if NRZI is not set. (See "Serial I/O Mode Register," for NRZI encoding mode format.)

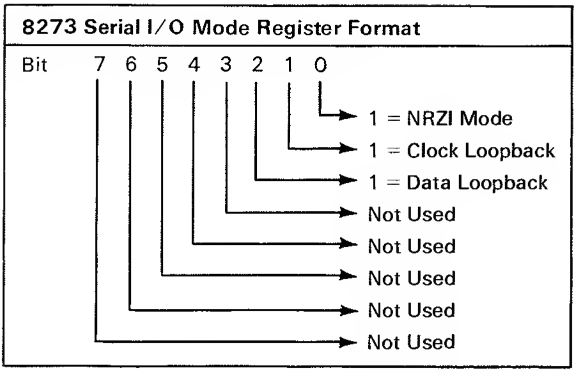
Bit 2 If bit 2 is set to a 1, the 8273 buffers the first two bytes of a received frame (the bytes are not passed to memory). Resetting this bit (to 0) causes these bytes to be passed to and from memory.

Bit 3 This bit indicates to the 8273 when to generate an end-of-frame interrupt. If bit 3 is set, an early interrupt is generated when the last data character has been passed to the 8273. If the processor responds to the early interrupt with another transmit command before the final flag is sent, the final-flag interrupt will not be generated and a new frame will begin when the current frame is complete. Thus, frames may be sent separated by a single flag. A reset condition causes an interrupt to be generated only following a final flag.

Bit 4 This is the EOP-interrupt-mode function and is not used on the SDLC communications adapter. This bit should always be in the reset condition.

Bit 5 This bit is always reset for SDLC operation, which causes the 8273 protocol controller to recognize eight ones (0 1 1 1 1 1 1 1) as an abort character.

Serial I/O Mode Register

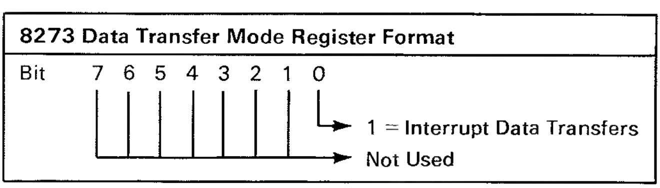


Bit 0 Set to 1, this bit specifies NRZI encoding and decoding. Resetting this bit specifies that transmit and receive data be treated as a normal positive-logic bit stream.

Bit 1 When bit 1 is set to 1, the transmit clock is internally routed to the receive-clock circuitry. It is normally used with the loopback bit (bit 2). The reset condition causes the transmit and receive clocks to be routed to their respective 8273 I/O pins.

Bit 2 When bit 2 is set, the transmitted data is internally routed to the received data circuitry. The reset condition causes the transmitted and received data to be routed to their respective 8273 I/O pins.

Data Transfer Mode Register



When the data transfer mode register is set, the 8273 protocol controller will interrupt when data bytes are required for transmission, or are available from a reception. If a transmit or receive interrupt occurs and the status register indicates that there is no transmit or receive interrupt result, the interrupt is a transmit or receive data request, respectively. Reset of this register causes DMA requests to be performed with no interrupts to the processor.

One-Bit Delay Mode Register

8273 One-Bit Delay Mode Register Format

```
Bit   7  6  5  4  3  2  1  0
      |  |  |  |  |  |  |  |
      |  └──┴──┴──┴──┴──┴──┴──► Not Used
      |
      └───────────────────────► 1 = One-Bit Delay Enable
```

When one-bit delay is set, the 8273 retransmits the received data stream one-bit delayed. Reset of this bit stops the one-bit delay mode.

The table below is a summary of all set and reset commands associated with the 8273 mode registers. The set or reset mask used to define individual bits is treated as a single parameter. No result or interrupt is generated by the 8273 after execution of these commands.

Register	Command	Hex Code	Parameter
One-Bit Delay Mode	Set	A4	Set Mask
	Reset	64	Reset Mask
Data Transfer Mode	Set	97	Set Mask
	Reset	57	Reset Mask
Operating Mode	Set	91	Set Mask
	Reset	51	Reset Mask
Serial I/O Mode	Set	A0	Set Mask
	Reset	60	Reset Mask

8273 SDLC Protocol Controller Mode Register Commands

Command Phase

Although the 8273 is a full duplex device, there is only one command register. Thus, the command register must be used for only one command sequence at a time and the transmitter and receiver may never be simultaneously in a command phase.

The system software starts the command phase by selecting the 8273 command register address and writing a command byte into the register. The following table lists command and parameter information for the 8273 protocol controller. If further information is required by the 8273 prior to execution of the command, the system software must write this information into the parameter register.

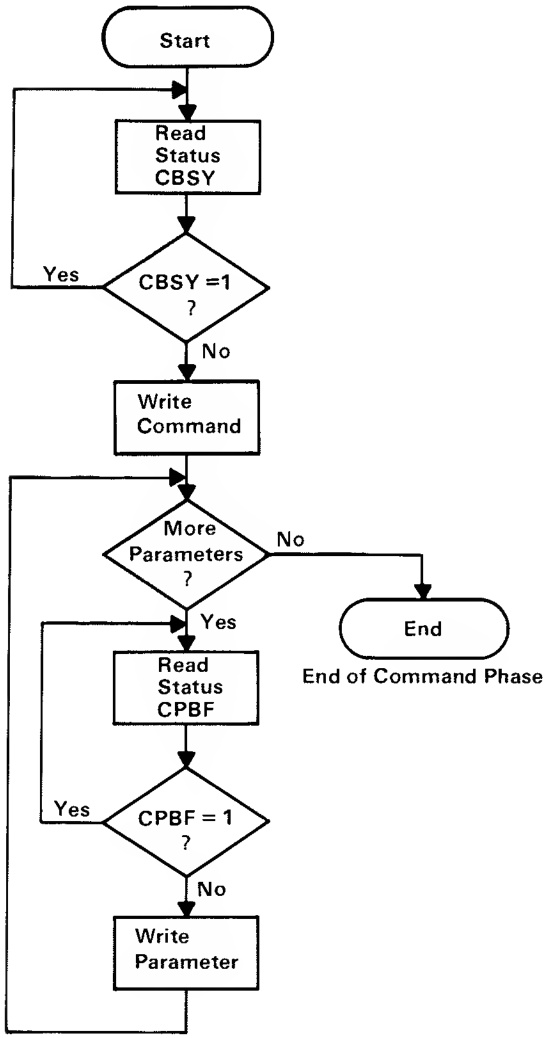
Command Description	Command (Hex)	Parameter	Results	Result Port	Completion Interrupt
Set One-Bit Delay	A4	Set Mask	None	—	No
Reset One-Bit Delay	64	Reset Mask	None	—	No
Set Data Transfer Mode	97	Set Mask	None	—	No
Reset Data Transfer Mode	57	Reset Mask	None	—	No
Set Operating Mode	91	Set Mask	None	—	No
Reset Operating Mode	51	Reset Mask	None	—	No
Set Serial I/O Mode	A0	Set Mask	None	—	No
Reset Serial I/O Mode	60	Reset Mask	None	—	No
General Receive	C0	80,81	RIC,R0,R1, A,C	RXI/R	Yes
Selective Receive	C1	80,81,A1, A2	RIC,R0,R1, A,C	RXI/R	Yes
Receive Disable	C5	None	None	—	No
Transmit Frame	C8	L0,L1,A,C	TIC	TXI/R	Yes
Transmit Transparent	C9	L0,L1	TIC	TXI/R	Yes
Abort Transmit Frame	CC	None	TIC	TXI/R	Yes
Abort Transmit Transparent	CD	None	TIC	TXI/R	Yes
Read Port A	22	None	Port Value	Result	No
Read Port B	23	None	Port Value	Result	No
Set Port B Bit	A3	Set Mask	None	—	No
Reset Port B Bit	63	Reset Mask	None	—	No

8273 Command Summary Key

B0 — Least significant byte of the receiver buffer length.
B1 — Most significant byte of the receiver buffer length.
L0 — Least significant byte of the Tx frame length.
L1 — Most significant byte of the Tx frame length.
A1 — Receive frame address match field one.
A2 — Receive frame address match field two.
A — Address field of received frame. If non-buffered mode is specified, this result is not provided.
C — Control field of received frame. If non-buffered mode is specified, this result is not provided.
RXI/R — Receive interrupt result register.
TXI/R — Transmit interrupt result register.
R0 — Least significant byte of the length of the frame received.
R1 — Most significant byte of the length of the frame received.
RIC — Receiver interrupt result code.
TIC — Transmitter interrupt result code.

8273 SDLC Protocol Controller Commands

A flowchart of the command phase is shown below. Handshaking of the command and parameter bytes is accomplished by the CBSY and CPBF bits of the status register. A command may not be written if the 8273 is busy (CBSY = 1). The original command will be overwritten if a second command is issued while CBSY = 1. The flowchart also indicates a parameter buffer full check. The processor must wait until CPBF = 0 before writing a parameter to the parameter register. Previous parameters are overwritten and lost if a parameter is written while CPBF = 1.



8273 SDLC Protocol Controller Command Phase Flowchart

Execution Phase

During the execution phase, the operation specified by the command phase is performed. If DMA is utilized for data transfers, no processor involvement is required.

For interrupt-driven transfers the 8273 raises the appropriate INT pin (TxINT or RxINT). When the processor responds to the interrupt, it must determine the cause by examining the status register and the associated IRA (interrupt result available) bit of the status register. If IRA = 0, the interrupt is a data transfer request. If IRA = 1, an operation is complete and the associated interrupt result register must be read to determine completion status.

Result Phase

During the result phase, the 8273 notifies the processor of the outcome of a command execution. This phase is initiated by either a successful completion or error detection during execution.

Some commands such as reading or writing the I/O ports provide immediate results. These results are made available to the processor in the 8273 result register. Presence of a valid immediate result is indicated by the CRBF (command result buffer full) bit of the status register.

Non-immediate results deal with the transmitter and receiver. These results are provided in the TxI/R (transmit interrupt result) or RxI/R (receiver interrupt result) registers, respectively. The 8273 notifies the processor that a result is available with the TxIRA and RxIRA bits of the status register. Results consist of one-byte result interrupt code indicating the condition for the interrupt and, if required, one or more bytes supplying additional information. The "Result Code Summary" table later in this section provides information on the format and decode of the transmitter and receiver results.

The following are typical frame transmit and receive sequences. These examples assume DMA is utilized for data transfer operations.

Transmit

Before a frame can be transmitted, the DMA controller is supplied, by the communication software, the starting address for the desired information field. The 8273 is then commanded to transmit a frame (by issuing a transmit frame command).

After a command, but before transmission begins, the 8273 needs some more information (parameters). Four parameters are required for the transmit frame command; the frame address field byte, the frame control field byte, and two bytes which are the least significant and most significant bytes of the information field byte length. Once all four parameters are loaded, the 8273 makes RTS (request to send) active and waits for CTS (clear to send) to go active from the modem interface. Once CTS is active, the 8273 starts the frame transmission. While the 8273 is transmitting the opening flag, address field, and control field, it starts making transmitter DMA requests. These requests continue at character (byte) boundaries until the pre-loaded number of bytes of information field have been transmitted. At this point, the requests stop, the FCS (frame check sequence) and closing flag are transmitted, and the TxINT line is raised, signaling the processor the frame transmission is complete and the result should be read. Note that after the initial command and parameter loading, no processor intervention was required (since DMA is used for data transfers) until the entire frame was transmitted.

General Receive

Receiver operation is very similar. Like the initial transmit sequence, the processor's DMA controller is loaded with a starting address for a receive data buffer and the 8273 is commanded to receive. Unlike the transmitter, there are two different receive commands; a general receive, where all received frames are transferred to memory, and selective receive, where only frames having an address field matching one of two preprogrammed 8273 address fields are transferred to memory.

(This example covers a general receive operation.) After the receive command, two parameters are required before the receiver becomes active; the least significant and most significant bytes of the receiver buffer length. Once these bytes are loaded, the receiver is active and the processor may return to other tasks. The next frame appearing at the receiver input is transferred to memory using receiver DMA requests. When the closing flag is received, the 8273 checks the FCS and raises its RxINT line. The processor can then read the results, which indicate if the frame was error-free or not. (If the received frame had been longer than the pre-loaded buffer length, the processor would have been notified of that occurrence earlier with a receiver error interrupt. Like the transmit example, after the initial command, the processor is free for other tasks until a frame is completely received.

Selective Receive

In selective receive, two parameters (A1 and A2) are required in addition to those for general receive. These parameters are two address match bytes. When commanded to selective receive, the 8273 passes to memory or the processor only those frames having an address field matching either A1 or A2. This command is usually used for secondary stations with A1 designating the secondary address and A2 being the "all parties" address. If only one match byte is needed, A1 and A2 should be equal. As in general receive, the 8273 counts the incoming data bytes and interrupts the processor if the received frame is larger than the preset receive buffer length.

Result Code Summary

	Hex Code	Result	Status After Interrupt
Transmit	0C	Early Transmit Interrupt	Transmitter Active
	0D	Frame Transmit Complete	Idle or Flags
	0E	DMA Underrun	Abort
	0F	Clear to Send Error	Abort
	10	Abort Complete	Idle or Flags
Receive	X0	A1 Match or General Receive	Active
	X1	A2 Match	Active
	03	CRC Error	Active
	04	Abort Detected	Active
	05	Idle Detected	Disabled
	06	EOP Detected	Disabled
	07	Frame Less Than 32 Bits	Active
	08	DMA Overrun	Disabled
	09	Memory Buffer Overflow	Disabled
	0A	Carrier Detect Failure	Disabled
	0B	Receiver Interrupt Overrun	Disabled

Note: X decodes to number of bits in partial byte received.

The first two codes in the receive result code table result from the error free reception of a frame. Since SDLC allows frames of arbitrary length (>32 bits), the high order bits of the receive result report the number of valid received bits in the last received information field byte. The chart below shows the decode of this receive result bit.

X	Bits Received in Last Byte
E	All Eight Bits of Last Byte
0	Bit0 Only
8	Bit1-Bit0
4	Bit2-Bit0
C	Bit3-Bit0
2	Bit4-Bit0
A	Bit5-Bit0
6	Bit6-Bit0

Address and Interrupt Information

The following tables provide address and interrupt information for the SDLC adapter:

Hex Code	Device	Register Name	Function
380	8255	Port A Data	Internal/External Sensing
381	8255	Port B Data	External Modem Interface
382	8255	Port C Data	Internal Control
383	8255	Mode Set	8255 Mode Initialization
384	8253	Counter 0 LSB	Square Wave Generator
384	8253	Counter 0 MSB	Square Wave Generator
385	8253	Counter 1 LSB	Inactivity Time-Outs
385	8253	Counter 1 MSB	Inactivity Time-Outs
386	8253	Counter 2 LSB	Inactivity Time-Outs
386	8253	Counter 2 MSB	Inactivity Time-Outs
387	8253	Mode Register	8253 Mode Set
388	8273	Command/Status	Out=Command In=Status
389	8273	Parameter/Result	Out=Parameter In=Status
38A	8273	Transmit INT Status	DMA/INT
38B	8273	Receive INT Status	DMA/INT
38C	8273	Data	DPC (Direct Program Control)

SDLC Communications Adapter Device Addresses

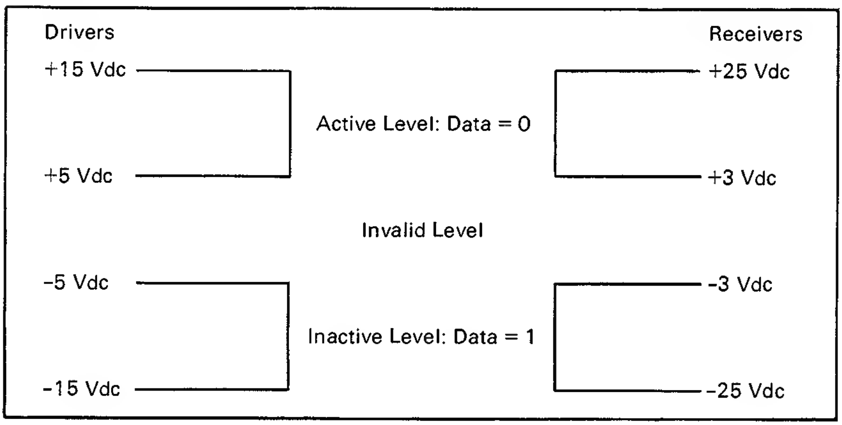
Interrupt Level 3	Transmit/Receive Interrupt
Interrupt Level 4	Timer 1 Interrupt Timer 2 Interrupt Clear to Send Changed Data Set Ready Changed
DMA Level One is used for Transmit and Receive	

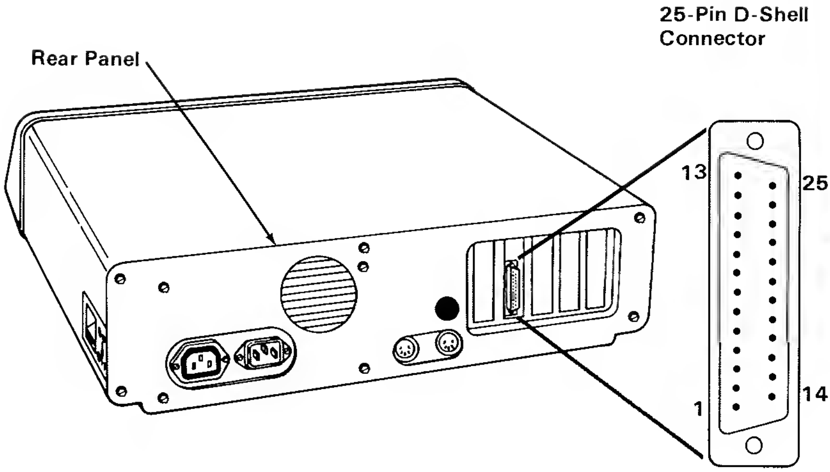
Interrupt Information

Interface Information

The SDLC communications adapter conforms to interface signal levels standardized by the Electronics Industries Association RS-232C Standard. These levels are shown in the figure below.

Additional lines used but not standardized by EIA are pins 11, 18, and 25. These lines are designated as select standby, test and test indicate, respectively. Select Standby is used to support the switched network backup facility of a modem providing this option. Test and test indicate support a modem wrap function on modems which are designed for business machine controlled modem wraps. Two jumpers on the adapter (P1 and P2) are used to connect test and test indicate to the interface, if required (see Appendix D for these jumpers).





External Device — Synchronous Data Link Control Communications Adapter

Signal Name — Description	Pin
No Connection	1
Transmitted Data	2
Received Data	3
Request to Send	4
Clear to Send	5
Data Set Ready	6
Signal Ground	7
Received Line Signal Detector	8
No Connection	9
No Connection	10
Select Standby*	11
No Connection	12
No Connection	13
No Connection	14
Transmitter Signal Element Timing	15
No Connection	16
Receiver Signal Element Timing	17
Test (IBM Modems Only)*	18
No Connection	19
Data Terminal Ready	20
No Connection	21
Ring Indicator	22
Data Signal Rate Selector	23
No Connection	24
Test Indicate (IBM Modems Only)*	25

*Not standardized by EIA (Electronics Industry Association).

Connector Specifications

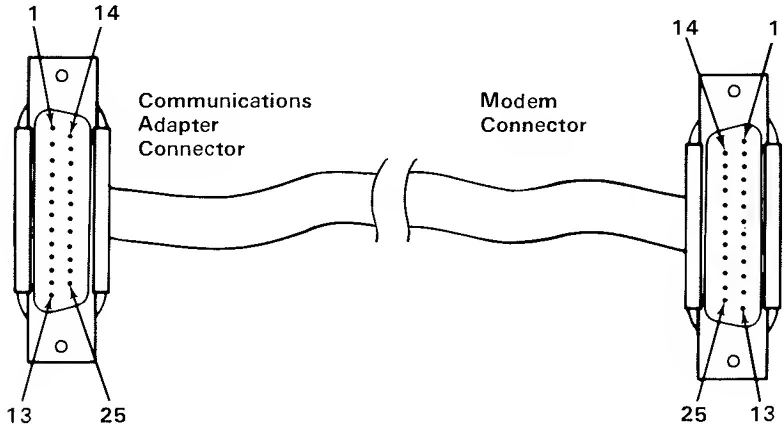
Notes:

IBM Communications Adapter Cable

The IBM Communications Adapter Cable is a ten foot cable for connection of an IBM communications adapter to a modem or other RS-232C DCE (data communications equipment). It is fully shielded and provides a high quality, low noise channel for interface between the communications adapter and DCE.

The connector ends are 25-pin D-shell connectors. All pin connections conform with the EIA RS-232C standard. In addition, connection is provided on pins 11, 18 and 25. These pins are designated as select standby, test and test indicate, respectively, on some modems. Select standby is used to support the switched network backup facility, if applicable. Test and test indicate support a modem wrap function on modems designed for business machine controlled modem wraps.

The IBM Communications Adapter Cable connects the following pins on the 25-pin D-shell connectors.



Communications Adapter Connector Pin #	Name	Modem Connector Pin #
NC	Outer Cable Shield	1
2	Transmitted Data	2
3	Received Data	3
4	Request to Send	4
5	Clear to Send	5
6	Data Set Ready	6
7	Signal Ground (Inner Lead Shields)	7
8	Received Line Signal Detector	8
NC		NC
NC		NC
11	Select Standby	11
NC		NC
NC		NC
NC		NC
15	Transmitter Signal Element Timing	15
NC		NC
17	Receiver Signal Element Timing	17
18	Test	18
NC		NC
20	Data Terminal Ready	20
NC		NC
22	Ring Indicator	22
23	Data Signal Rate Selector	23
NC		NC
25	Test Indicate	25

Connector Specifications

SECTION 2: ROM BIOS AND SYSTEM USAGE

ROM BIOS 2-2
Keyboard Encoding and Usage 2-11
BIOS Cassette Logic 2-21

BIOS

Notes:

ROM BIOS

BIOS

The basic input/output system (BIOS) resides in ROM on the system board and provides device level control for the major I/O devices in the system. Additional ROM modules may be located on option adapters to provide device level control for that option adapter. BIOS routines enable the assembly language programmer to perform block (disk and diskette) or character-level I/O operations without concern for device address and operating characteristics. System services, such as time-of-day and memory size determination, are provided by the BIOS.

The goal is to provide an operational interface to the system and relieve the programmer of the concern about the characteristics of hardware devices. The BIOS interface insulates the user from the hardware, thus allowing new devices to be added to the system, yet retaining the BIOS level interface to the device. In this manner, user programs become transparent to hardware modifications and enhancements.

The IBM Personal Computer MACRO Assembler manual and the IBM Personal Computer Disk Operating System (DOS) manual provide useful programming information related to this section. A complete listing of the BIOS is given in Appendix A.

Use of BIOS

Access to BIOS is through the 8088 software interrupts. Each BIOS entry point is available through its own interrupt, which can be found in the "8088 Software Interrupt Listing."

The software interrupts, hex 10 through hex 1A, each access a different BIOS routine. For example, to determine the amount of memory available in the system,

```
INT   12H
```

will invoke the BIOS routine for determining memory size and will return the value to the caller.

Parameter Passing

All parameters passed to and from the BIOS routines go through the 8088 registers. The prolog of each BIOS function indicates the registers used on the call and the return. For the memory size example, no parameters are passed. The memory size, in 1K byte increments, is returned in the AX register.

If a BIOS function has several possible operations, the AH register is used at input to indicate the desired operation. For example, to set the time of day, the following code is required:

```
MOV  AH,1                ;function is to set time of day.
MOV  CX,HIGH_COUNT       ;establish the current time.
MOV  DX,LOW_COUNT
INT  1AH                 ;set the time.
```

To read the time of day:

```
MOV  AH,0                ;function is to read time of
                         day.
INT  1AH                 ;read the timer.
```

Generally, the BIOS routines save all registers except for AX and the flags. Other registers are modified on return only if they are returning a value to the caller. The exact register usage can be seen in the prolog of each BIOS function.

Address (Hex)	Interrupt Number	Name	BIOS Entry
0-3	0	Divide by Zero	D_EOI
4-7	1	Single Step	D_EOI
8-B	2	Nonmaskable	NMI_INT
C-F	3	Breakpoint	D_EOI
10-13	4	Overflow	D_EOI
14-17	5	Print Screen	PRINT_SCREEN
18-1B	6	Reserved	D_EOI
1D-1F	7	Reserved	D_EOI
20-23	8	Time of Day	TIMER_INT
24-27	9	Keyboard	KB_INT
28-2B	A	Reserved	D_EOI
2C-2F	B	Communications	D_EOI
30-33	C	Communications	D_EOI
34-37	D	Disk	D_EOI
38-3B	E	Diskette	DISK_INT
3C-3F	F	Printer	D_EOI
40-43	10	Video	VIDEO_IO
44-47	11	Equipment Check	EQUIPMENT
48-4B	12	Memory	MEMORY_SIZE_DETERMINE
4C-4F	13	Diskette/Disk	DISKETTE_IO
50-53	14	Communications	RS232_IO
54-57	15	Cassette	CASSETTE_IO
58-5B	16	Keyboard	KEYBOARD_IO
5C-5F	17	Printer	PRINTER_IO
60-63	18	Resident BASIC	F600:0000
64-67	19	Bootstrap	BOOT_STRAP
68-6B	1A	Time of Day	TIME_OF_DAY
6C-6F	1B	Keyboard Break	DUMMY_RETURN
70-73	1C	Timer Tick	DUMMY_RETURN
74-77	1D	Video Initialization	VIDEO_PARMS
78-7B	1E	Diskette Parameters	DISK_BASE
7C-7F	1F	Video Graphics Chars	0

8088 Software Interrupt Listing

Vectors with Special Meanings

Interrupt Hex 1B – Keyboard Break Address

This vector points to the code to be exercised when the Ctrl and Break keys are pressed on the keyboard. The vector is invoked while responding to the keyboard interrupt, and control should be returned through an IRET instruction. The power-on routines initialize this vector to point to an IRET instruction, so that nothing will occur when the Ctrl and Break keys are pressed unless the application program sets a different value.

Control may be retained by this routine, with the following problems. The Break may have occurred during interrupt processing, so that one or more End of Interrupt commands must be sent to the 8259 controller. Also, all I/O devices should be reset in case an operation was underway at that time.

Interrupt Hex 1C – Timer Tick

This vector points to the code to be executed on every system-clock tick. This vector is invoked while responding to the timer interrupt, and control should be returned through an IRET instruction. The power-on routines initialize this vector to point to an IRET instruction, so that nothing will occur unless the application modifies the pointer. It is the responsibility of the application to save and restore all registers that will be modified.

Interrupt Hex 1D – Video Parameters

This vector points to a data region containing the parameters required for the initialization of the 6845 on the video card. Note that there are four separate tables, and all four must be reproduced if all modes of operation are to be supported. The power-on routines initialize this vector to point to the parameters contained in the ROM video routines.

Interrupt Hex 1E – Diskette Parameters

This vector points to a data region containing the parameters required for the diskette drive. The power-on routines initialize the vector to point to the parameters contained in the ROM diskette routine. These default parameters represent the specified values for any IBM drives attached to the machine. Changing this parameter block may be necessary to reflect the specifications of the other drives attached.

Interrupt Hex 1F – Graphics Character Extensions

When operating in the graphics modes of the IBM Color/Graphics Monitor Adapter (320 by 200 or 640 by 200), the read/write character interface will form the character from the ASCII code point, using a set of dot patterns. The dot patterns for the first 128 code points are contained in ROM. To access the second 128 code points, this vector must be established to point at a table of up to 1K bytes, where each code point is represented by eight bytes of graphic information. At power-on, this vector is initialized to 000:0, and it is the responsibility of the user to change this vector if the additional code points are required.

Interrupt Hex 40 – Reserved

When an IBM Fixed Disk Drive Adapter is installed, the BIOS routines use interrupt hex 40 to revector the diskette pointer.

Interrupt Hex 41 – Fixed Disk Parameters

This vector points to a data region containing the parameters required for the fixed disk drive. The power-on routines initialize the vector to point to the parameters contained in the ROM disk routine. These default parameters represent the specified values for any IBM Fixed Disk Drives attached to the machine. Changing this parameter block may be necessary to reflect the specifications of the other fixed disk drives attached.

Other Read/Write Memory Usage

The IBM BIOS routines use 256 bytes of memory starting at absolute hex 400 to hex 4FF. Locations hex 400 to 407 contain the base addresses of any RS-232C cards attached to the system. Locations hex 408 to 40F contain the base addresses of the printer adapter.

Memory locations hex 300 to 3FF are used as a stack area during the power-on initialization, and bootstrap, when control is passed to it from power-on. If the user desires the stack in a different area, the area must be set by the application.

Address (Hex)	Interrupt (Hex)	Function
80-83	20	DOS Program Terminate
84-87	21	DOS Function Call
88-8B	22	DOS Terminate Address
8C-8F	23	DOS Ctrl Break Exit Address
90-93	24	DOS Fatal Error Vector
94-97	25	DOS Absolute Disk Read
98-9B	26	DOS Absolute Disk Write
9C-9F	27	DOS Terminate, Fix In Storage
A0-FF	28-3F	Reserved for DOS
100-17F	40-5F	Reserved
180-19F	60-67	Reserved for User Software Interrupts
1A0-1FF	68-7F	Not Used
200-217	80-85	Reserved by BASIC
218-3C3	86-F0	Used by BASIC Interpreter while BASIC is running
3C4-3FF	F1-FF	Not Used

BASIC and DOS Reserved Interrupts

Address (Hex)	Mode	Function
400-48F	ROM BIOS	See BIOS Listing
490-4EF		Reserved
4F0-4FF		Reserved as Intra-Application Communication Area for any application
500-5FF		Reserved for DOS and BASIC
500	DOS	Print Screen Status Flag Store 0-Print Screen Not Active or Successful Print Screen Operation 1-Print Screen In Progress 255-Error Encountered during Print Screen Operation
504	DOS	Single Drive Mode Status Byte
510-511	BASIC	BASIC's Segment Address Store
512-515	BASIC	Clock Interrupt Vector Segment: Offset Store
516-519	BASIC	Break Key Interrupt Vector Segment: Offset Store
51A-51D	BASIC	Disk Error Interrupt Vector Segment: Offset Store

Reserved Memory Locations

If you do DEF SEG (Default workspace segment):

	Offset (Hex Value)	Length
Line number of current line being executed	2E	2
Line number of last error	347	2
Offset into segment of start of program text	30	2
Offset into segment of start of variables (end of program text 1-1)	358	2
Keyboard buffer contents if 0-no characters in buffer if 1-characters in buffer	6A	1
Character color in graphics mode Set to 1, 2, or 3 to get text in colors 1 to 3. Do not set to 0. (Default = 3)	4E	1

Example

```
100 Print  PEEK (&H2E) + 256*PEEK (&H2F)
                 L            H
  100      | Hex 64  | Hex 00 |
```

BASIC Workspace Variables

Starting Address in Hex	
00000	BIOS Interrupt Vectors
00080	Available Interrupt Vectors
00400	BIOS Data Area
00500	User Read/Write Memory
C8000	Disk Adapter
F0000	Read Only Memory
FE000	Bios Program Area

BIOS Memory Map

BIOS Programming Hints

The BIOS code is invoked through software interrupts. The programmer should not "hard code" BIOS addresses into applications. The internal workings and absolute addresses within BIOS are subject to change without notice.

If an error is reported by the disk or diskette code, you should reset the drive adapter and retry the operation. A specified number of retries should be required on diskette reads to ensure the problem is not due to motor start-up.

When altering I/O port bit values, the programmer should change only those bits which are necessary to the current task. Upon completion, the programmer should restore the original environment. Failure to adhere to this practice may be incompatible with present and future applications.

Adapter Cards with System-Accessible ROM Modules

The ROM BIOS provides a facility to integrate adapter cards with on board ROM code into the system. During the POST, interrupt vectors are established for the BIOS calls. After the default vectors are in place, a scan for additional ROM modules takes place. At this point, a ROM routine on the adapter card may gain control. The routine may establish or intercept interrupt vectors to hook themselves into the system.

The absolute addresses hex C8000 through hex F4000 are scanned in 2K blocks in search of a valid adapter card ROM. A valid ROM is defined as follows:

Byte 0:	Hex 55
Byte 1:	Hex AA
Byte 2:	A length indicator representing the number of 512 byte blocks in the ROM (length/512). A checksum is also done to test the integrity of the ROM module. Each byte in the defined ROM is summed modulo hex 100. This sum must be 0 for the module to be deemed valid.

When the POST identifies a valid ROM, it does a far call to byte 3 of the ROM (which should be executable code). The adapter card may now perform its power-on initialization tasks. The feature ROM should return control to the BIOS routines by executing a far return.

Notes:

Keyboard Encoding and Usage

Encoding

The keyboard routine provided by IBM in the ROM BIOS is responsible for converting the keyboard scan codes into what will be termed "Extended ASCII."

Extended ASCII encompasses one-byte character codes with possible values of 0 to 255, an extended code for certain extended keyboard functions, and functions handled within the keyboard routine or through interrupts.

Character Codes

The following character codes are passed through the BIOS keyboard routine to the system or application program. A "−1" means the combination is suppressed in the keyboard routine. The codes are returned in AL. See Appendix C for the exact codes. Also, see "Keyboard Scan Code Diagram" in Section 1.

Key Number	Base Case	Upper Case	Ctrl	Alt
1	Esc	Esc	Esc	-1
2	1	!	-1	Note 1
3	2	@	Nul (000) Note 1	Note 1
4	3	#	-1	Note 1
5	4	$	-1	Note 1
6	5	%	-1	Note 1
7	6	^	RS(030)	Note 1
8	7	&	-1	Note 1
9	8	*	-1	Note 1
10	9	(	-1	Note 1
11	0	)	-1	Note 1
12	-	—	US(031)	Note 1
13	=	+	-1	Note 1
14	Backspace (008)	Backspace (008)	Del (127)	-1
15	→| (009)	|← (Note 1)	-1	-1
16	q	Q	DC1 (017)	Note 1
17	w	W	ETB (023)	Note 1

Character Codes (Part 1 of 3)

BIOS

Key Number	Base Case	Upper Case	Ctrl	Alt
18	e	E	ENQ (005)	Note 1
19	r	R	DC2 (018)	Note 1
20	t	T	DC4 (020)	Note 1
21	y	Y	EM (025)	Note 1
22	u	U	NAK (021)	Note 1
23	i	I	HT (009)	Note 1
24	o	O	SI (015)	Note 1
25	p	P	DLE (016)	Note 1
26	[	{	Esc (027)	-1
27	]	}	GS (029	-1
28	CR	CR	LF (010)	-1
29 Ctrl	-1	-1	-1	-1
30	a	A	SOH (001)	Note 1
31	s	S	DC3 (019)	Note 1
32	d	D	EOT (004)	Note 1
33	f	F	ACK (006)	Note 1
34	g	G	BEL (007)	Note 1
35	h	H	BS (008)	Note 1
36	j	J	LF (010)	Note 1
37	k	K	VT (011)	Note 1
38	l	L	FF (012)	Note 1
39	;	:	-1	-1
40	'	"	-1	-1
41	`	~	-1	-1
42 Shift	-1	-1	-1	-1
43	\	¦	FS (028)	-1
44	z	Z	SUB (026)	Note 1
45	x	X	CAN (024)	Note 1
46	c	C	ETX (003)	Note 1
47	v	V	SYN (022)	Note 1
48	b	B	STX (002)	Note 1
49	n	N	SO (014)	Note 1
50	m	M	CR (013)	Note 1
51	,	<	-1	-1
52	.	>	-1	-1
53	/	?	-1	-1
54 Shift	-1	-1	-1	-1
55	*	(Note 2)	(Note 1)	-1
56 Alt	-1	-1	-1	-1
57	SP	SP	SP	SP
58 Caps Lock	-1	-1	-1	-1
59	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
60	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
61	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
62	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
63	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
64	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)

Character Codes (Part 2 of 3)

Key Number	Base Case	Upper Case	Ctrl	Alt
65	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
66	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
67	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
68	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
69 Num Lock	-1	-1	Pause (Note 2)	-1
70 Scroll Lock	-1	-1	Break (Note 2)	-1

Notes: 1. Refer to "Extended Codes" in this section.
2. Refer to "Special Handling" in this section.

Character Codes (Part 3 of 3)

Keys 71 to 83 have meaning only in base case, in Num Lock (or shifted) states, or in Ctrl state. It should be noted that the shift key temporarily reverses the current Num Lock state.

Key Number	Num Lock	Base Case	Alt	Ctrl
71	7	Home (Note 1)	-1	Clear Screen
72	8	↑ (Note 1)	-1	-1
73	9	Page Up (Note 1)	-1	Top of Text and Home
74	-	--------------------	-1	-1
75	4	← (Note 1)	-1	Reverse Word (Note 1)
76	5	-1	-1	-1
77	6	→ (Note 1)	-1	Advance Word (Note 1)
78	+	+	-1	-1
79	1	End (Note 1)	-1	Erase to EOL (Note 1)
80	2	↓ (Note 1)	-1	-1
81	3	Page Down (Note 1)	-1	Erase to EOS (Note 1)
82	0	Ins	-1	-1
83	.	Del (Notes 1,2)	Note 2	Note 2

Notes: 1. Refer to "Extended Codes" in this section.
2. Refer to "Special Handling" in this section.

BIOS

Extended Codes

Extended Functions

For certain functions that cannot be represented in the standard ASCII code, an extended code is used. A character code of 000 (Nul) is returned in AL. This indicates that the system or application program should examine a second code that will indicate the actual function. Usually, but not always, this second code is the scan code of the primary key that was pressed. This code is returned in AH.

Second Code	Function
3	Nul Character
15	←
16-25	Alt Q, W, E, R, T, Y, U, I, O, P
30-38	Alt A, S, D, F, G, H, J, K, L
44-50	Alt Z, X, C, V, B, N, M
59-68	F1 to F10 Function Keys Base Case
71	Home
72	↑
73	Page Up and Home Cursor
75	←
77	→
79	End
80	↓
81	Page Down and Home Cursor
82	Ins (Insert)
83	Del (Delete)
84-93	F11 to F20 (Uppercase F1 to F10)
94-103	F21 to F30 (Ctrl F1 to F10)
104-113	F31 to F40 (Alt F1 to F10)
114	Ctrl PrtSc (Start/Stop Echo to Printer)
115	Ctrl ← (Reverse Word)
116	Ctrl → (Advance Word)
117	Ctrl End [Erase to End of Line (EOL)]
118	Ctrl PgDn [Erase to End of Screen (EOS)]
119	Ctrl Home (Clear Screen and Home)
120-131	Alt 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, -, = (Keys 2-13)
132	Ctrl PgUp (Top 25 Lines of Text and Home Cursor)

Keyboard Extended Functions

Shift States

Most shift states are handled within the keyboard routine, transparent to the system or application program. In any case, the current set of active shift states are available by calling an entry point in the ROM keyboard routine. The following keys result in altered shift states:

Shift

This key temporarily shifts keys 2-13, 15-27, 30-41, 43-53, 55, and 59-68 to upper case (base case if in Caps Lock state). Also, the Shift key temporarily reverses the Num Lock or non-Num-Lock state of keys 71-73, 75, 77, and 79-83.

Ctrl

This key temporarily shifts keys 3, 7, 12, 14, 16-28, 30-38, 43-50, 55, 59-71, 73, 75, 77, 79, and 81 to the Ctrl state. Also, the Ctrl key is used with the Alt and Del keys to cause the "system reset" function, with the Scroll Lock key to cause the "break" function, and with the Num Lock key to cause the "pause" function. The system reset, break, and pause functions are described in "Special Handling" on the following pages.

Alt

This key temporarily shifts keys 2-13, 16-25, 30-38, 44-50, and 59-68 to the Alt state. Also, the Alt key is used with the Ctrl and Del keys to cause the "system reset" function described in "Special Handling" on the following pages.

The Alt key has another use. This key allows the user to enter any character code from 0 to 255 into the system from the keyboard. The user holds down the Alt key and types the decimal value of the characters desired using the numeric keypad (keys 71-73, 75-77, and 79-82). The Alt key is then released. If more than three digits are typed, a modulo-256 result is created. These three digits are interpreted as a character code and are transmitted through the keyboard routine to the system or application program. Alt is handled internal to the keyboard routine.

Caps Lock

This key shifts keys 16-25, 30-38, and 44-50 to upper case. A second depression of the Caps Lock key reverses the action. Caps Lock is handled internal to the keyboard routine.

Scroll Lock

This key is interpreted by appropriate application programs as indicating use of the cursor-control keys should cause windowing over the text rather than cursor movement. A second depression of the Scroll Lock key reverses the action. The keyboard routine simply records the current shift state of the Scroll Lock key. It is the responsibility of the system or application program to perform the function.

Shift Key Priorities and Combinations

If combinations of the Alt, Ctrl, and Shift keys are pressed and only one is valid, the precedence is as follows: the Alt key is first, the Ctrl key is second, and the Shift key is third. The only valid combination is Alt and Ctrl, which is used in the "system reset" function.

Special Handling

System Reset

The combination of the Alt, Ctrl, and Del keys will result in the keyboard routine initiating the equivalent of a "system reset" or "reboot." System reset is handled internal to the keyboard.

Break

The combination of the Ctrl and Break keys will result in the keyboard routine signaling interrupt hex 1A. Also, the extended characters (AL = hex 00, AH = hex 00) will be returned.

Pause

The combination of the Ctrl and Num Lock keys will cause the keyboard interrupt routine to loop, waiting for any key except the Num Lock key to be pressed. This provides a system- or application-transparent method of temporarily suspending list, print, and so on, and then resuming the operation. The "unpause" key is thrown away. Pause is handled internal to the keyboard routine.

Print Screen

The combination of the Shift and PrtSc (key 55) keys will result in an interrupt invoking the print screen routine. This routine works in the alphanumeric or graphics mode, with unrecognizable characters printing as blanks.

Other Characteristics

The keyboard routine does its own buffering. The keyboard buffer is large enough to support a fast typist. However, if a key is entered when the buffer is full, the key will be ignored and the "bell" will be sounded.

Also, the keyboard routine suppresses the typematic action of the following keys: Ctrl, Shift, Alt, Num Lock, Scroll Lock, Caps Lock, and Ins.

Keyboard Usage

This section is intended to outline a set of guidelines of key usage when performing commonly used functions.

Function	Key(s)	Comment
Home Cursor	Home	Editors; word processors
Return to outermost menu	Home	Menu driven applications
Move cursor up	↑	Full screen editor, word processor
Page up, scroll backward 25 lines and home	PgUp	Editors; word processors
Move cursor left	← Key 75	Text, command entry
Move cursor right	→	Text, command entry
Scroll to end of text Place cursor at end of line	End	Editors; word processors
Move cursor down	↓	Full screen editor, word processor
Page down, scroll forward 25 lines and home	Pg Dn	Editors; word processors
Start/Stop insert text at cursor, shift text right in buffer	Ins	Text, command entry
Delete character at cursor	Del	Text, command entry
Destructive backspace	← Key 14	Text, command entry
Tab forward	→|	Text entry
Tab reverse	|←	Text entry
Clear screen and home	Ctrl Home	Command entry
Scroll up	↑	In scroll lock mode
Scroll down	↓	In scroll lock mode
Scroll left	←	In scroll lock mode
Scroll right	→	In scroll lock mode
Delete from cursor to EOL	Ctrl End	Text, command entry
Exit/Escape	Esc	Editor, 1 level of menu, and so on
Start/Stop Echo screen to printer	Ctrl Prt Sc (Key 55)	Any time
Delete from cursor to EOS	Ctrl PgDn	Text, command entry
Advance word	Ctrl →	Text entry
Reverse word	Ctrl ←	Text entry
Window Right	Ctrl →	When text is too wide to fit screen
Window Left	Ctrl ←	When text is too wide to fit screen
Enter insert mode	Ins	Line editor

Keyboard - Commonly Used Functions (Part 1 of 2)

Function	Key(s)	Comment
Exit insert mode	Ins	Line editor
Cancel current line	Esc	Command entry, text entry
Suspend system (pause)	Ctrl Num Lock	Stop list, stop program, and so on Resumes on any key
Break interrupt	Ctrl Break	Interrupt current process
System reset	Alt Ctrl Del	Reboot
Top of document and home cursor	Ctrl PgUp	Editors, word processors
Standard function keys	F1-F10	Primary function keys
Secondary function keys	Shift F1-F10 Ctrl F1-F10 Alt F1-F10	Extra function keys if 10 are not sufficient
Extra function keys	Alt Keys 2-13 (1-9,0,-,=)	Used when stickers are put along top of keyboard
Extra function keys	Alt A-Z	Used when function starts with same letter as one of the alpha keys

Keyboard - Commonly Used Functions (Part 2 of 2)

Function	Key
Carriage return	↵
Line feed	Ctrl ↵
Bell	Ctrl G
Home	Home
Cursor up	↑
Cursor down	↓
Cursor left	←
Cursor right	→
Advance one word	Ctrl →
Reverse one word	Ctrl ←
Insert	Ins
Delete	Del
Clear screen	Ctrl Home
Freeze output	Ctrl Num Lock
Tab advance	⇥
Stop execution (break)	Ctrl Break
Delete current line	Esc
Delete to end of line	Ctrl End
Position cursor to end of line	End

BASIC Screen Editor Special Functions

Function	Key
Suspend	Ctrl Num Lock
Echo to printer	Ctrl PrtSc (Key 55 any case)
Stop echo to printer	Ctrl PrtSc (Key 55 any case)
Exit current function (break)	Ctrl Break
Backspace	← Key 14
Line feed	Ctrl ↵
Cancel line	Esc
Copy character	F1 or →
Copy until match	F2
Copy remaining	F3
Skip character	Del
Skip until match	F4
Enter skip mode	Ins
Exit insert mode	Ins
Make new line the template	F5
String separator in REPLACE	F6
End of file in keyboard input	F6

DOS Special Functions

BIOS Cassette Logic

Software Algorithms – Interrupt Hex 15

The cassette routine will be called by the request type in AH. The address of the bytes to be read from or written to the tape will be specified by ES:BX and the number of bytes to be read or written will be specified by CX. The actual number of bytes read will be returned in DX. The read block and write block will automatically turn the cassette motor on at the start and off at the end. The request types in AH and the cassette status descriptions follow:

Request Type	Function
AH = 0	Turn Cassette Motor On
AH = 1	Turn Cassette Motor Off
AH = 2	Read Tape Block Read CX bytes into memory starting at Address ES:BX Return actual number of bytes read in DX Return Cassette Status in AH
AH = 3	Write Tape Block Write CX bytes onto cassette starting at Address DS:BX Return Cassette Status in AH

Cassette Status	Description
AH = 00	No Errors
AH = 01	Cyclic Redundancy Check (CRC) Error in Read Block
AH = 02	No Data Transitions
AH = 04	No Leader
AH = 80	Invalid Command

Note: The carry flag will be set on any error.

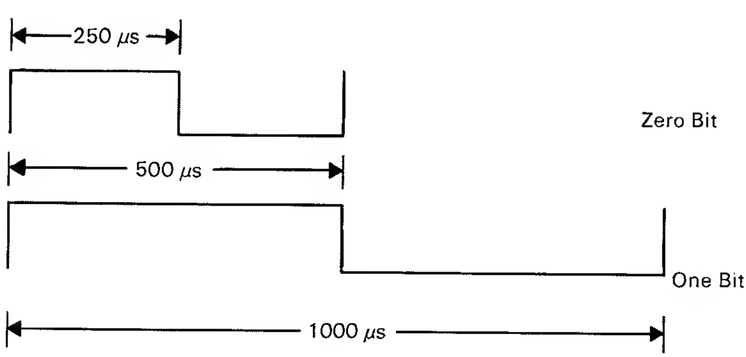
Cassette Write

The write-block routine writes a tape block onto the cassette tape. The block is described in "Data Record Architecture" later in this section.

The write-block routine turns on the cassette drive motor and a synchronization bit (0) and then writes the leader (256 bytes of all 1's) to the tape. Next, the routine writes the number of data blocks specified by CX. After each data block of 256 bytes, a 2-byte cyclic redundancy check (CRC) is written. The data bytes are taken from the memory location pointed at by ES.

The write-byte routine disassembles and writes the byte a bit at a time to the cassette. The method used is to set Timer 2 to the period of the desired data bit. The timer is set to a period of 1.0 millisecond for a 1 bit and 0.5 millisecond for a 0 bit.

The timer is set to mode 3, which means the timer outputs a square wave with a period given by its counter register. The timer's period is changed on the fly for each data bit written to the cassette. If the number of data bytes to be written is not an integral multiple of 256, then, after the last desired data byte from memory has been written, the data block is extended to 256 bytes of writing multiples of the last data byte. The last block is closed with two CRC bytes as usual. After the last data block, a trailer consisting of four bytes of all 1 bits is written. Finally, the cassette motor is turned off, if there are no errors reported by the routine.



Cassette Read

The read-block routine turns on the cassette drive motor and then delays for approximately 0.5 second to allow the motor to come up to speed.

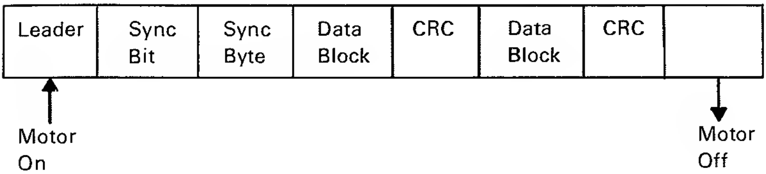
The read-block routine then searches for the leader and must detect all 1 bits for approximately 1/4 of the leader length before it can look for the sync (0) bit. After the sync bit is detected, the sync byte (ASCII character hex 16) is read. If the sync byte is read correctly, the data portion can be read. If a correct sync byte is not found, the routine goes back and searches for the leader again. The data is read a bit at a time and assembled into bytes. After each byte is assembled, it is written into memory at location ES:BX and BX is incremented by 1.

After each multiple of 256 data bytes is read, the CRC is read and compared to the CRC generated. If a CRC error is detected, the routine will exit with the carry flag set to indicate an error and the status of AH set to hex 01. DX will contain the number of bytes written memory.

The time of day interrupt (IRQ0) is disabled during the cassette-read operation.

Data Record Architecture

The write-block routine uses the following format to record a tape block onto a cassette tape:



Component	Description
Leader	256 Bytes (of All 1's)
Sync Bit	One 0 Bit
Sync Byte	ASCII Character Hex 16
Data Blocks	256 Bytes in Length
CRC	2 Bytes for each Data Block

Data Record Components

Error Recovery

Error recovery is handled through software. A CRC is used to detect errors. The polynomial used is $G(X) = X^{16} + X^{12} + X^{5} + 1$, which is the polynomial used by the synchronous data link control interface. Essentially, as bits are written to or read from the cassette tape, they are passed through the CRC register in software. After a block of data is written, the complemented value of the calculcated CRC register is written on the tape. Upon reading the cassette data, the CRC bytes are read and compared to the generated CRC value. If the read CRC does not equal the generated CRC, the processor's carry flag is set and the status of AH is set to hex 01, which indicates a CRC error has occurred. Also, the routine is exited on a CRC error.

APPENDIX A: ROM BIOS LISTINGS

	Page	Line Number
System ROM BIOS		
Equates	A-2	12
8088 Interrupt Locations	A-2	34
Stack	A-2	66
Data Areas	A-2	74
Power-On Self-Test	A-5	229
Boot Strap Loader	A-21	1493
I/O Support		
Asynchronous Communications (RS-232C)	A-22	1551
Keyboard	A-26	1818
Diskette	A-36	2426
Printer	A-46	3201
Display	A-47	3327
System Configuration Analysis		
Memory Size Determination	A-73	5177
Equipment Determination	A-73	5208
Cassette I/O Support	A-74	5253
Graphics Character Generator	A-80	5769
Time of Day	A-82	5903
Print Screen	A-84	6077
Fixed Disk ROM BIOS		
Fixed Disk I/O Interface	A-87	1
Boot Strap Loader	A-92	399

```
LOC OBJ          SOURCE

                 $TITLE(BIOS FOR IBM PERSONAL COMPUTER)

                 ;-------------------------------------------------------------
                 ;       THE  BIOS ROUTINES ARE MEANT TO BE ACCESSED THROUGH    :
                 ;       SOFTWARE INTERRUPTS ONLY.  ANY ADDRESSES PRESENT IN    :
                 ;       THE LISTINGS  ARE INCLUDED  ONLY FOR  COMPLETENESS,    :
                 ;       NOT FOR  REFERENCE.   APPLICATIONS WHICH  REFERENCE    :
                 ;       ABSOLUTE   ADDRESSES   WITHIN   THE  CODE   SEGMENT    :
                 ;       VIOLATE THE STRUCTURE AND DESIGN OF BIOS.              :
                 ;-------------------------------------------------------------

                 ;--------------------------------------
                 ;               EQUATES               :
                 ;--------------------------------------
 0060            PORT_A          EQU     60H             ; 8255 PORT A ADDR
 0061            PORT_B          EQU     61H             ; 8255 PORT B ADDR
 0062            PORT_C          EQU     62H             ; 8255 PORT C ADDR
 0063            CMD_PORT        EQU     63H
 0020            INTA00          EQU     20H             ; 8259 PORT
 0021            INTA01          EQU     21H             ; 8259 PORT
 0020            EOI             EQU     20H
 0040            TIMER           EQU     40H
 0043            TIM_CTL         EQU     43H             ; 8253 TIMER CONTROL PORT ADDR
 0040            TIMER0          EQU     40H             ; 8253 TIMER/CNTER 0 PORT ADDR
 0001            TMINT           EQU     01              ; TIMER 0 INTR RECVD MASK
 0008            DMA08           EQU     08              ; DMA STATUS REG PORT ADDR
 0000            DMA             EQU     00              ; DMA CHANNEL 0 ADDR REG PORT ADDR
 0540            MAX_PERIOD      EQU     540H
 0410            MIN_PERIOD      EQU     410H
 0060            KBD_IN          EQU     60H             ; KEYBOARD DATA IN ADDR PORT
 0002            KBDINT          EQU     02              ; KEYBOARD INTR MASK
 0060            KB_DATA         EQU     60H             ; KEYBOARD SCAN CODE PORT
 0061            KB_CTL          EQU     61H             ; CONTROL BITS FOR KB SENSE DATA
                 ;--------------------------------------
                 ;       8088 INTERRUPT LOCATIONS      :
                 ;--------------------------------------
----             ABS0            SEGMENT AT 0
0000             STG_LOC0        LABEL   BYTE
0008                             ORG     2*4
0008             NMI_PTR         LABEL    WORD
0014                             ORG     5*4
0014             INT5_PTR        LABEL   WORD
0020                             ORG     8*4
0020             INT_ADDR        LABEL   WORD
0020             INT_PTR         LABEL   DWORD
0040                             ORG     10H*4
0040             VIDEO_INT       LABEL   WORD
0074                             ORG     1DH*4
0074             PARM_PTR        LABEL   DWORD           ; POINTER TO VIDEO PARMS
0060                             ORG     18H*4
0060             BASIC_PTR       LABEL   WORD            ; ENTRY POINT FOR CASSETTE BASIC
0078                             ORG     01EH*4          ; INTERRUPT 1EH
0078             DISK_POINTER    LABEL   DWORD
007C                             ORG     01FH*4          ; LOCATION OF POINTER
007C             EXT_PTR LABEL   DWORD                   ; POINTER TO EXTENSION
0100                             ORG     040H*4          ; ROUTINE
0100 ????        IO_ROM_INIT     DW      ?               ;
0102 ????        IO_ROM_SEG      DW      ?               ; OPTIONAL ROM SEGMENT
0400                             ORG     400H
0400             DATA_AREA       LABEL   BYTE            ; ABSOLUTE LOCATION OF DATA SEGMENT
0400             DATA_WORD       LABEL   WORD
7C00                             ORG     7C00H
7C00             BOOT_LOCN       LABEL   FAR
----             ABS0            ENDS

                 ;--------------------------------------------------------
                 ;       STACK -- USED DURING INITIALIZATION ONLY        :
                 ;--------------------------------------------------------
----             STACK           SEGMENT AT 30H
0000 (128                        DW      128 DUP(?)
     ????
     )
0100             TOS             LABEL   WORD
----             STACK           ENDS

                 ;--------------------------------------
                 ;       ROM BIOS DATA AREAS           :
                 ;--------------------------------------
----             DATA            SEGMENT AT 40H
```

```
LOC OBJ                LINE   SOURCE

0000 (4                       RS232_BASE      DW      4 DUP(?)        ; ADDRESSES OF RS232 ADAPTERS
      ????
      )
0008 (4                       PRINTER_BASE    DW      4 DUP(?)        ; ADDRESSES OF PRINTERS
      ????
      )
0010 ????                     EQUIP_FLAG      DW      ?               ; INSTALLED HARDWARE
0012 ??                       MFG_TST         DB      ?               ; INITIALIZATION FLAG
0013 ????                     MEMORY_SIZE     DW      ?               ; MEMORY SIZE IN K BYTES
0015 ????                     IO_RAM_SIZE     DW      ?               ; MEMORY IN I/O CHANNEL
                              ;-----------------------------------------
                              ;            KEYBOARD DATA AREAS         :
                              ;-----------------------------------------
0017 ??                       KB_FLAG         DB      ?

                              ;----- SHIFT FLAG EQUATES WITHIN KB_FLAG

  0080                        INS_STATE       EQU     80H             ; INSERT STATE IS ACTIVE
  0040                        CAPS_STATE      EQU     40H             ; CAPS LOCK STATE HAS BEEN TOGGLED
  0020                        NUM_STATE       EQU     20H             ; NUM LOCK STATE HAS BEEN TOGGLED
  0010                        SCROLL_STATE    EQU     10H             ; SCROLL LOCK STATE HAS BEEN TOGGLED
  0008                        ALT_SHIFT       EQU     08H             ; ALTERNATE SHIFT KEY DEPRESSED
  0004                        CTL_SHIFT       EQU     04H             ; CONTROL SHIFT KEY DEPRESSED
  0002                        LEFT_SHIFT      EQU     02H             ; LEFT SHIFT KEY DEPRESSED
  0001                        RIGHT_SHIFT     EQU     01H             ; RIGHT SHIFT KEY DEPRESSED

0018 ??                       KB_FLAG_1       DB      ?               ; SECOND BYTE OF KEYBOARD STATUS

  0080                        INS_SHIFT       EQU     80H             ; INSERT KEY IS DEPRESSED
  0040                        CAPS_SHIFT      EQU     40H             ; CAPS LOCK KEY IS DEPRESSED
  0020                        NUM_SHIFT       EQU     20H             ; NUM LOCK KEY IS DEPRESSED
  0010                        SCROLL_SHIFT    EQU     10H             ; SCROLL LOCK KEY IS DEPRESSED
  0008                        HOLD_STATE      EQU     08H             ; SUSPEND KEY HAS BEEN TOGGLED

0019 ??                       ALT_INPUT       DB      ?               ; STORAGE FOR ALTERNATE KEYPAD ENTRY
001A ????                     BUFFER_HEAD     DW      ?               ; POINTER TO HEAD OF KEYBOARD BUFFER
001C ????                     BUFFER_TAIL     DW      ?               ; POINTER TO TAIL OF KEYBOARD BUFFER
001E (16                      KB_BUFFER       DW      16 DUP(?)       ; ROOM FOR 15 ENTRIES
      ????
      )
003E                          KB_BUFFER_END   LABEL   WORD

                              ;----- HEAD = TAIL INDICATES THAT THE BUFFER IS EMPTY

  0045                        NUM_KEY         EQU     69              ; SCAN CODE FOR NUMBER LOCK
  0046                        SCROLL_KEY      EQU     70              ; SCROLL LOCK KEY
  0038                        ALT_KEY         EQU     56              ; ALTERNATE SHIFT KEY SCAN CODE
  001D                        CTL_KEY         EQU     29              ; SCAN CODE FOR CONTROL KEY
  003A                        CAPS_KEY        EQU     58              ; SCAN CODE FOR SHIFT LOCK
  002A                        LEFT_KEY        EQU     42              ; SCAN CODE FOR LEFT SHIFT
  0036                        RIGHT_KEY       EQU     54              ; SCAN CODE FOR RIGHT SHIFT
  0052                        INS_KEY         EQU     82              ; SCAN CODE FOR INSERT KEY
  0053                        DEL_KEY         EQU     83              ; SCAN CODE FOR DELETE KEY

                              ;-----------------------------------------
                              ;            DISKETTE DATA AREAS         :
                              ;-----------------------------------------
003E ??                       SEEK_STATUS     DB      ?               ; DRIVE RECALIBRATION STATUS
                              ;                       BIT 3-0 = DRIVE 3-0 NEEDS RECAL BEFORE
                              ;                                 NEXT SEEK IF BIT IS = 0
  0080                        INT_FLAG        EQU     080H            ; INTERRUPT OCCURRENCE FLAG
003F ??                       MOTOR_STATUS    DB      ?               ; MOTOR STATUS
                              ;                       BIT 3-0 = DRIVE 3-0 IS CURRENTLY RUNNING
                              ;                       BIT 7   = CURRENT OP IS A WRITE, REQUIRES DELAY
0040 ??                       MOTOR_COUNT     DB      ?               ; TIME OUT COUNTER FOR DRIVE TURN OFF
  0025                        MOTOR_WAIT      EQU     37              ; TWO SEC OF COUNT FOR MOTOR TURN OFF

0041 ??                       DISKETTE_STATUS DB      ?               ; BYTE OF RETURN CODE INFO FOR STATUS
  0080                        TIME_OUT        EQU     80H             ; ATTACHMENT FAILED TO RESPOND
  0040                        BAD_SEEK        EQU     40H             ; SEEK OPERATION FAILED
  0020                        BAD_NEC         EQU     20H             ; NEC CONTROLLER HAS FAILED
  0010                        BAD_CRC         EQU     10H             ; BAD CRC ON DISKETTE READ
  0009                        DMA_BOUNDARY    EQU     09H             ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
  0008                        BAD_DMA         EQU     08H             ; DMA OVERRUN ON OPERATION
  0004                        RECORD_NOT_FND  EQU     04H             ; REQUESTED SECTOR NOT FOUND
  0003                        WRITE_PROTECT   EQU     03H             ; WRITE ATTEMPTED ON WRITE PROT DISK
  0002                        BAD_ADDR_MARK   EQU     02H             ; ADDRESS MARK NOT FOUND
```

Appendix A

```
LOC OBJ                 SOURCE

  0001                  BAD_CMD         EQU     01H             ; BAD COMMAND PASSED TO DISKETTE I/O

0042 (7                 NEC_STATUS      DB      7 DUP(?)        ; STATUS BYTES FROM NEC
     ??
     )

                        ;----------------------------------------
                        ;       VIDEO DISPLAY DATA AREA         :
                        ;----------------------------------------
0049 ??                 CRT_MODE        DB      ?               ; CURRENT CRT MODE
004A ????               CRT_COLS        DW      ?               ; NUMBER OF COLUMNS ON SCREEN
004C ????               CRT_LEN         DW      ?               ; LENGTH OF REGEN IN BYTES
004E ????               CRT_START       DW      ?               ; STARTING ADDRESS IN REGEN BUFFER
0050 (8                 CURSOR_POSN     DW      8 DUP(?)        ; CURSOR FOR EACH OF UP TO 8 PAGES
     ????
     )
0060 ????               CURSOR_MODE     DW      ?               ; CURRENT CURSOR MODE SETTING
0062 ??                 ACTIVE_PAGE     DB      ?               ; CURRENT PAGE BEING DISPLAYED
0063 ????               ADDR_6845       DW      ?               ; BASE ADDRESS FOR ACTIVE DISPLAY CARD
0065 ??                 CRT_MODE_SET    DB      ?               ; CURRENT SETTING OF THE 3X8 REGISTER
0066 ??                 CRT_PALETTE     DB      ?               ; CURRENT PALETTE SETTING COLOR CARD

                        ;----------------------------------------
                        ;          CASSETTE DATA AREA           :
                        ;----------------------------------------
0067 ????               EDGE_CNT        DW      ?               ; TIME COUNT AT DATA EDGE
0069 ????               CRC_REG         DW      ?               ; CRC REGISTER
006B ??                 LAST_VAL        DB      ?               ; LAST INPUT VALUE

                        ;----------------------------------------
                        ;            TIMER DATA AREA            :
                        ;----------------------------------------
006C ????               TIMER_LOW       DW      ?               ; LOW WORD OF TIMER COUNT
006E ????               TIMER_HIGH      DW      ?               ; HIGH WORD OF TIMER COUNT
0070 ??                 TIMER_OFL       DB      ?               ; TIMER HAS ROLLED OVER SINCE LAST READ
                        ;COUNTS_SEC     EQU     18
                        ;COUNTS_MIN     EQU     1092
                        ;COUNTS_HOUR    EQU     65543
                        ;COUNTS_DAY     EQU     1573040 = 1800B0H

                        ;----------------------------------------
                        ;           SYSTEM DATA AREA            :
                        ;----------------------------------------
0071 ??                 BIOS_BREAK      DB      ?               ; BIT 7 = 1 IF BREAK KEY WAS DEPRESSED
0072 ????               RESET_FLAG      DW      ?               ; WORD = 1234H IF KB RESET UNDERWAY
                        ;----------------------------------------
                        ;          FIXED DISK DATA AREA         :
                        ;----------------------------------------
0074 ????                               DW      ?               ;
0076 ????                               DW      ?               ;
                        ;----------------------------------------
                        ;     PRINTER AND RS232 TIMEOUT CTRS    :
                        ;----------------------------------------
0078 (4                 PRINT_TIM_OUT   DB      4 DUP(?)        ; PRINTER TIME OUT COUNTER
     ??
     )
007C (4                 RS232_TIM_OUT   DB      4 DUP(?)        ; RS232 TIME OUT COUNTER
     ??
     )
                        ;----------------------------------------
                        ;       EXTRA KEYBOARD DATA AREA        :
                        ;----------------------------------------
0080 ????               BUFFER_START    DW      ?
0082 ????               BUFFER_END      DW      ?
----                    DATA    ENDS
                        ;----------------------------------------
                        ;           EXTRA DATA AREA             :
                        ;----------------------------------------
----                    XXDATA          SEGMENT AT 50H
0000 ??                 STATUS_BYTE     DB      ?
----                    XXDATA          ENDS

                        ;----------------------------------------
                        ;         VIDEO DISPLAY BUFFER          :
                        ;----------------------------------------
----                    VIDEO_RAM       SEGMENT AT 0B800H
```

```
LOC OBJ                 SOURCE

0000                    REGEN           LABEL   BYTE
0000                    REGENW          LABEL   WORD
0000 (16384                             DB      16384 DUP(?)
      ??
      )
----                    VIDEO_RAM       ENDS
                        ;---------------------------------------
                        ;           ROM RESIDENT CODE           :
                        ;---------------------------------------
----                    CODE            SEGMENT AT 0F000H
0000 (57344                             DB      57344 DUP(?)            ; FILL LOWEST 56K
      ??
      )

E000 31353031343736                     DB      '1501476 COPR. IBM 1981'        ; COPYRIGHT NOTICE
     20434F50522E20
     49424D20313938
     32

                        ;-------------------------------------------------------------
                        ;       INITIAL RELIABILITY TESTS -- PHASE 1                 :
                        ;-------------------------------------------------------------
                                        ASSUME  CS:CODE,SS:CODE,ES:ABS0,DS:DATA
                        ;---------------------------------------
                        ;       DATA DEFINITIONS                :
                        ;---------------------------------------
E016 D1E0               C1      DW      C11             ; RETURN ADDRESS

                        ;----------------------------------------------------------------------
                        ;       THIS SUBROUTINE PERFORMS A READ/WRITE STORAGE TEST ON         :
                        ;       A 16K BLOCK OF STORAGE.                                       :
                        ; ENTRY REQUIREMENTS:                                                 :
                        ;       ES = ADDRESS OF STORAGE SEGMENT BEING TESTED                  :
                        ;       DS = ADDRESS OF STORAGE SEGMENT BEING TESTED                  :
                        ;       WHEN ENTERING AT STGTST_CNT, CX MUST BE LOADED WITH           :
                        ;       THE BYTE COUNT.                                               :
                        ; EXIT PARAMETERS:                                                    :
                        ;       ZERO FLAG = 0 IF STORAGE ERROR (DATA COMPARE OR PARITY CHECK. :
                        ;               AL = 0 DENOTES A PARITY CHECK. ELSE AL=XOR'ED BIT     :
                        ;                       PATTERN OF THE EXPECTED DATA PATTERN VS THE   :
                        ;                       ACTUAL DATA READ.                             :
                        ;       AX,BX,CX,DX,DI, AND SI ARE ALL DESTROYED.                     :
                        ;----------------------------------------------------------------------

E018                    STGTST  PROC    NEAR
E018 B90040                     MOV     CX,4000H                ; SETUP CNT TO TEST A 16K BLK
E01B                    STGTST_CNT:
E01B FC                         CLD                             ; SET DIR FLAG TO INCREMENT
E01C 8BD9                       MOV     BX,CX                   ; SAVE BYTE CNT (4K FOR VIDEO OR 16K)
E01E B8AAAA                     MOV     AX,0AAAAH               ; GET DATA PATTERN TO WRITE
E021 BA55FF                     MOV     DX,0FF55H               ; SETUP OTHER DATA PATTERNS TO USE
E024 2BFF                       SUB     DI,DI                   ; DI = OFFSET 0 RELATIVE TO ES REG
E026 F3                         REP     STOSB                   ; WRITE STORAGE LOCATIONS
E027 AA
E028                    C3:                                     ; STG01
E028 4F                         DEC     DI                      ; POINT TO LAST BYTE JUST WRITTEN
E029 FD                         STD                             ; SET DIR FLAG TO GO BACKWARDS
E02A                    C4:
E02A 8BF7                       MOV     SI,DI
E02C 8BCB                       MOV     CX,BX                   ; SETUP BYTE CNT
E02E                    C5:                                     ; INNER TEST LOOP
E02E AC                         LODSB                           ; READ OLD TST BYTE FROM STORAGE [SI]+
E02F 32C4                       XOR     AL,AH                   ; DATA READ AS EXPECTED ?
E031 7525                       JNE     C7                      ;  NO - GO TO ERROR ROUTINE
E033 8AC2                       MOV     AL,DL                   ; GET NEXT DATA PATTERN TO WRITE
E035 AA                         STOSB                           ; WRITE INTO LOCATION JUST READ [DI]+
E036 E2F6                       LOOP    C5                      ;  DECREMENT BYTE COUNT AND LOOP CX

E038 22E4                       AND     AH,AH                   ; ENDING ZERO PATTERN WRITTEN TO STG ?
E03A 7416                       JZ      C6X                     ; YES - RETURN TO CALLER WITH AL=0
E03C 8AE0                       MOV     AH,AL                   ; SETUP NEW VALUE FOR COMPARE
E03E 86F2                       XCHG    DH,DL                   ; MOVE NEXT DATA PATTERN TO DL
E040 22E4                       AND     AH,AH                   ; READING ZERO PATTERN THIS PASS ?
E042 7504                       JNZ     C6                      ; CONTINUE TEST SEQUENCE TILL ZERO DATA
E044 8AD4                       MOV     DL,AH                   ;  ELSE SET ZERO FOR END READ PATTERN
E046 EBE0                       JMP     C3                      ;  AND MAKE FINAL BACKWARDS PASS
E048                    C6:
```

Appendix A

```
LOC OBJ        SOURCE

E048 FC                CLD                        ; SET OIR FLAG TO 60 FORWARO
E049 47                INC     OI                 ; SET POINTER TO BEG LOCATION
E04A 740E              JZ      C4                 ; READ/WRITE FORWARD IN STG
E04C 4F                DEC     DI                 ; ADJUST POINTER
E040 BA0100            MOV     OX,00001H          ; SETUP 01 FOR PARITY BIT
                                                  ;  AND 00 FOR END
E050 EB06              JMP     C3                 ; REAO/WRITE BACKWARD IN STG
E052           C6X:
E052 E462              IN      AL,PORT_C          ; OIO A PARITY ERROR OCCUR ?
E054 24C0              AND     AL,0C0H            ; ZERO FLAG WILL BE OFF PARITY ERROR
E056 B000              MOV     AL,000H            ; AL=0 OATA COMPARE OK
E058           C7:
E058 FC                CLD                        ; SET OEFAULT OIRCTN FLAG BACK TO INC
E059 C3                RET
               STGTST  ENDP
               ;-----------------------------------------------------------
               ;       8088 PROCESSOR TEST                                :
               ; DESCRIPTION                                              :
               ;       VERIFY 8088 FLAGS, REGISTERS AND CONDITIONAL JUMPS :
               ;-----------------------------------------------------------
                       ASSUME  CS:CODE,OS:NOTHING,ES:NOTHING,SS:NOTHING
E05B                   ORG     0E05BH
E05B           RESET   LABEL   FAR
E05B           START:
E05B FA                CLI                        ; OISABLE INTERRUPTS
E05C B4D5              MOV     AH,005H            ; SET SF, CF, ZF, AND AF FLAGS ON
E05E 9E                SAHF
E05F 734C              JNC     ERR01              ; GO TO ERR ROUTINE IF CF NOT SET
E061 754A              JNZ     ERR01              ; GO TO ERR ROUTINE IF ZF NOT SET
E063 7B48              JNP     ERR01              ; GO TO ERR ROUTINE IF PF NOT SET
E065 7946              JNS     ERR01              ; GO TO ERR ROUTINE IF SF NOT SET
E067 9F                LAHF                       ; LOAD FLAG IMAGE TO AH
E068 B105              MOV     CL,5               ; LOAD CNT REG WITH SHIFT CNT
E06A D2EC              SHR     AH,CL              ; SHIFT AF INTO CARRY BIT POS
E06C 733F              JNC     ERR01              ; GO TO ERR ROUTINE IF AF NOT SET
E06E B040              MOV     AL,40H             ; SET THE OF FLAG ON
E070 D0E0              SHL     AL,1               ; SETUP FOR TESTING
E072 7139              JNO     ERR01              ; GO TO ERR ROUTINE IF OF NOT SET
E074 32E4              XOR     AH,AH              ; SET AH = 0
E076 9E                SAHF                       ; CLEAR SF, CF, ZF, AND PF
E077 7634              JBE     ERR01              ; GO TO ERR ROUTINE IF CF ON
                                                  ; OR TO TO ERR ROUTINE IF ZF ON
E079 7832              JS      ERR01              ; GO TO ERR ROUTINE IF SF ON
E07B 7A30              JP      ERR01              ; GO TO ERR ROUTINE IF PF ON
E07D 9F                LAHF                       ; LOAD FLAG IMAGE TO AH
E07E B105              MOV     CL,5               ; LOAD CNT REG WITH SHIFT CNT
E080 D2EC              SHR     AH,CL              ; SHIFT 'AF' INTO CARRY BIT POS
E082 7229              JC      ERR01              ; GO TO ERR ROUTINE IF ON
E084 D0E4              SHL     AH,1               ; CHECK THAT 'OF' IS CLEAR
E086 7025              JO      ERR01              ; GO TO ERR ROUTINE IF ON

               ;----- READ/WRITE THE 8088 GENERAL AND SEGMENTATION REGISTERS
               ;       WITH ALL ONE'S AND ZEROES'S.

E088 B8FFFF            MOV     AX,0FFFFH          ; SETUP ONE'S PATTERN IN AX
E08B F9                STC
E08C           C8:
E08C 8ED8              MOV     DS,AX              ; WRITE PATTERN TO ALL REGS
E08E 8CDB              MOV     BX,OS
E090 8EC3              MOV     ES,BX
E092 8CC1              MOV     CX,ES
E094 8ED1              MOV     SS,CX
E096 8CD2              MOV     OX,SS
E098 8BE2              MOV     SP,DX
E09A 8BEC              MOV     BP,SP
E09C 8BF5              MOV     SI,BP
E09E 8BFE              MOV     DI,SI
E0A0 7307              JNC     C9                 ; TST1A
E0A2 33C7              XOR     AX,OI              ; PATTERN MAKE IT THRU ALL REGS
E0A4 7507              JNZ     ERR01              ; NO - 60 TO ERR ROUTINE
E0A6 F8                CLC
E0A7 EBE3              JMP     C8
E0A9           C9:                                ; TST1A
E0A9 0BC7              OR      AX,OI              ; ZERO PATTERN MAKE IT THRU?
E0AB 7401              JZ      C10                ; YES - 60 TO NEXT TEST
E0AD F4        ERR01:  HLT                        ; HALT SYSTEM
               ;--------------------------------------------------
```

```
LOC OBJ                    SOURCE

                    ;       ROS CHECKSUM TEST I                                :
                    ; DESCRIPTION                                              :
                    ;       A CHECKSUM IS DONE FOR THE 8K ROS MODULE           :
                    ;       CONTAINING POD AND BIOS.                           :
                    ;-----------------------------------------------------------
E0AE                C10:
                                                        ; ZERO IN AL ALREADY
E0AE E6A0                   OUT     0A0H,AL             ; DISABLE NMI INTERRUPTS
E0B0 E683                   OUT     83H,AL              ; INITIALZE DMA PAGE REG
E0B2 BAD803                 MOV     DX,3D8H
E0B5 EE                     OUT     DX,AL               ; DISABLE COLOR VIDEO
E0B6 FEC0                   INC     AL
E0B8 B2B8                   MOV     DL,0B8H
E0BA EE                     OUT     DX,AL               ; DISABLE B/W VIDEO,EN HIGH RES
E0BB B099                   MOV     AL,99H              ; SET 8255 A,C-INPUT,B-OUTPUT
E0BD E663                   OUT     CMD_PORT,AL         ; WRITE 8255 CMD/MODE REG
E0BF B0FC                   MOV     AL,0FCH             ; DISABLE PARITY CHECKERS AND
E0C1 E661                   OUT     PORT_B,AL           ;  GATE SNS SWS,CASS MOTOR OFF
E0C3 8CC8                   MOV     AX,CS               ; SETUP SS SEG REG
E0C5 8ED0                   MOV     SS,AX
E0C7 8ED8                   MOV     DS,AX               ; SET UP DATA SEG TO POINT TO
                                                        ;  ROM ADDRESS
                            ASSUME  SS:CODE
E0C9 B7E0                   MOV     BH,0E0H             ; SETUP STARTING ROS ADDR (E0000)
E0CB BC16E0                 MOV     SP,OFFSET C1        ; SETUP RETURN ADDRESS
E0CE E97B0B                 JMP     ROS_CHECKSUM
E0D1                C11:
E0D1 750A                   JNE     ERR01               ; HALT SYSTEM IF ERROR
                    ;-------------------------------------------------------------------
                    ;       8237 DMA INITIALIZATION CHANNEL REGISTER TEST              :
                    ; DESCRIPTION                                                      :
                    ;       DISABLE THE 8237 DMA CONTROLLER.  VERIFY THAT TIMER 1      :
                    ;       FUNCTIONS OK. WRITE/READ THE CURRENT ADDRESS AND WORD      :
                    ;       COUNT REGISTERS FOR ALL CHANNELS.  INITIALIZE AND          :
                    ;       START DMA FOR MEMORY REFRESH.                              :
                    ;-------------------------------------------------------------------
E0D3 B004                   MOV     AL,04               ; DISABLE DMA CONTROLLER
E0D5 E608                   OUT     DMA08,AL

                    ;----- VERIFY THAT TIMER 1 FUNCTIONS OK

E0D7 B054                   MOV     AL,54H              ; SEL TIMER 1,LSB,MODE 2
E0D9 E643                   OUT     TIMER+3,AL
E0DB 8AC1                   MOV     AL,CL               ; SET INITIAL TIMER CNT TO 0
E0DD E641                   OUT     TIMER+1,AL
E0DF                C12:                                ; TIMER1_BITS_ON
E0DF B040                   MOV     AL,40H              ; LATCH TIMER 1 COUNT
E0E1 E643                   OUT     TIMER+3,AL
E0E3 80FBFF                 CMP     BL,0FFH             ; YES - SEE IF ALL BITS GO OFF
E0E6 7407                   JE      C13                 ; TIMER1_BITS_OFF
E0E8 E441                   IN      AL,TIMER+1          ; READ TIMER 1 COUNT
E0EA 0AD8                   OR      BL,AL               ; ALL BITS ON IN TIMER
E0EC E2F1                   LOOP    C12                 ; TIMER1_BITS_ON
E0EE F4                     HLT                         ; TIMER 1 FAILURE, HALT SYS
E0EF                C13:                                ; TIMER1_BITS_OFF
E0EF 8AC3                   MOV     AL,BL               ; SET TIMER 1 CNT
E0F1 2BC9                   SUB     CX,CX
E0F3 E641                   OUT     TIMER+1,AL
E0F5                C14:                                ; TIMER_LOOP
E0F5 B040                   MOV     AL,40H              ; LATCH TIMER 1 COUNT
E0F7 E643                   OUT     TIMER+3,AL
E0F9 90                     NOP                         ; DELAY FOR TIMER
E0FA 90                     NOP
E0FB E441                   IN      AL,TIMER+1          ; READ TIMER 1 COUNT
E0FD 22D8                   AND     BL,AL
E0FF 7403                   JZ      C15                 ; GO TO WRAP_DMA_REG
E101 E2F2                   LOOP    C14                 ; TIMER_LOOP
E103 F4                     HLT                         ; TIMER ERROR - HALT SYSTEM

                    ;----- INITIALIZE TIMER 1 TO REFRESH MEMORY

E104                C15:                                ; WRAP_DMA_REG
E104 B012                   MOV     AL,18               ; SETUP DIVISOR FOR REFRESH
E106 E641                   OUT     TIMER+1,AL          ; WRITE TIMER 1 CNT REG
E108 E60D                   OUT     DMA+0DH,AL          ; SEND MASTER CLEAR TO DMA
```

Appendix A

```
LOC OBJ                SOURCE

                       ;----- WRAP DMA CHANNELS ADDRESS AND COUNT REGISTERS

E10A B0FF                      MOV     AL,0FFH                 ; WRITE PATTERN FF TO ALL REGS
E10C                   C16:
E10C 8AD8                      MOV     BL,AL                   ; SAVE PATTERN FOR COMPARE
E10E 8AF8                      MOV     BH,AL
E110 B90800                    MOV     CX,8                    ; SETUP LOOP CNT
E113 2BD2                      SUB     DX,DX                   ; SETUP I/O PORT ADDR OF REG (0000)
E115                   C17:
E115 EE                        OUT     DX,AL                   ; WRITE PATTERN TO REG, LSB
E116 50                        PUSH    AX
E117 EE                        OUT     DX,AL                   ; MSB OF 16 BIT REG
E118 B80101                    MOV     AX,0101H                ; AX TO ANOTHER PAT BEFORE RD
E11B EC                        IN      AL,DX                   ; READ 16-BIT DMA CH REG, LSB
E11C 8AE0                      MOV     AH,AL                   ; SAVE LSB OF 16-BIT REG
E11E EC                        IN      AL,DX                   ; READ MSB OF DMA CH REG
E11F 3BD8                      CMP     BX,AX                   ; PATTERN READ AS WRITTEN?
E121 7401                      JE      C18                     ; YES - CHECK NEXT REG
E123 F4                        HLT                             ; NO - HALT THE SYSTEM
E124                   C18:                                    ; NXT_DMA_CH
E124 42                        INC     DX                      ; SET I/O PORT TO NEXT CH REG
E125 E2EE                      LOOP    C17                     ; WRITE PATTERN TO NEXT REG
E127 FEC0                      INC     AL                      ; SET PATTERN TO 0
E129 74E1                      JZ      C16                     ; WRITE TO CHANNEL REGS

                       ;----- INITIALIZE AND START DMA FOR MEMORY REFRESH.

E12B 8EDB                      MOV     DS,BX                   ; SET UP ABS0 INTO DS AND ES
E12D 8EC3                      MOV     ES,BX
                               ASSUME  DS:ABS0,ES:ABS0

E12F B0FF                      MOV     AL,0FFH                 ; SET CNT OF 64K FOR RAM REFRESH
E131 E601                      OUT     DMA+1,AL
E133 50                        PUSH    AX
E134 E601                      OUT     DMA+1,AL
E136 B20B                      MOV     DL,0BH                  ; DX=000B
E138 B058                      MOV     AL,058H                 ; SET DMA MODE,CH 0,READ,AUTOINT
E13A EE                        OUT     DX,AL                   ; WRITE DMA MODE REG
E13B B000                      MOV     AL,0                    ; ENABLE DMA CONTROLLER
E13D E608                      OUT     DMA+8,AL                ; SETUP DMA COMMAND REG
E13F 50                        PUSH    AX
E140 E60A                      OUT     DMA+10,AL               ; ENABLE CHANNEL 0 FOR REFRESH
E142 B103                      MOV     CL,3
E144 B041                      MOV     AL,41H                  ; SET MODE FOR CHANNEL 1
E146                   C18A:
E146 EE                        OUT     DX,AL
E147 FEC0                      INC     AL                      ; POINT TO NEXT CHANNEL
E149 E2FB                      LOOP    C18A
                       ;------------------------------------------------------------------
                       ;       BASE 16K READ/WRITE STORAGE TEST                          :
                       ; DESCRIPTION                                                     :
                       ;       WRITE/READ/VERIFY DATA PATTERNS FF,55,AA,01, AND 00       :
                       ;       TO 1ST 16K OF STORAGE.  VERIFY STORAGE ADDRESSABILITY.    :
                       ;       INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP FOR         :
                       ;       CHECKING MANUFACTURING TEST 2 MODE.                       :
                       ;------------------------------------------------------------------

                       ;----- DETERMINE MEMORY SIZE AND FILL MEMORY WITH DATA

E14B BA1302                    MOV     DX,0213H                ; ENABLE EXPANSION BOX
E14E B001                      MOV     AL,01H
E150 EE                        OUT     DX,AL
E151 8B2E7204                  MOV     BP,DATA_WORD[OFFSET RESET_FLAG] ; SAVE 'RESET_FLAG' IN BP
E155 81FD3412                  CMP     BP,1234H                ; WARM START?
E159 740A                      JE      C18B                    ; BYPASS STG TST.
E15B BC41F090                  MOV     SP,OFFSET C2
E15F E9B6FE                    JMP     STGTST
E162                   C24:
E162 7401                      JE      C18B                    ; PROCEED IF STGTST OK
E164 F4                        HLT                             ; HALT IF NOT
E165                   C18B:
E165 2BFF                      SUB     DI,DI
E167 E460                      IN      AL,PORT_A               ; DETERMINE BASE RAM SIZE
E169 240C                      AND     AL,0CH                  ; ISOLATE RAM SIZE SWS
E16B 0404                      ADD     AL, 4                   ; CALCULATE MEMORY SIZE
E16D B10C                      MOV     CL, 12
```

```
LOC OBJ                SOURCE

E16F D3E0                       SHL     AX, CL
E171 8BC8                       MOV     CX, AX
E173 FC                         CLD                             ; SET DIR FLAG TO INCR
E174                    C19:
E174 AA                         STOSB                           ; FILL BASE RAM WITH DATA
E175 E2FD                       LOOP    C19                     ; LOOP TIL ALL ZERO
E177 892E7204                   MOV     DATA_WORD[OFFSET RESET_FLAG],BP

                        ;----- DETERMINE IO CHANNEL RAM SIZE

E17B B0F8                       MOV     AL,0F8H                 ; ENABLE SWITCH 5
E17D E661                       OUT     PORT_B,AL
E17F E462                       IN      AL,PORT_C               ; READ SWITCHES
E181 2401                       AND     AL,00000001B            ; ISOLATE SWITCH 5
E183 B10C                       MOV     CL,12D
E185 D3C0                       ROL     AX,CL
E187 B0FC                       MOV     AL,0FCH                 ; DISABLE SW. 5
E189 E661                       OUT     PORT_B,AL
E18B E462                       IN      AL,PORT_C
E18D 240F                       AND     AL,0FH
E18F 0AC4                       OR      AL,AH                   ; COMBINE SWITCH VALUES
E191 8AD8                       MOV     BL,AL                   ; SAVE
E193 B420                       MOV     AH,32
E195 F6E4                       MUL     AH                      ; CALC. LENGTH
E197 A31504                     MOV     DATA_WORD[OFFSET IO_RAM_SIZE],AX        ;SAVE IT
E19A 7418                       JZ      C21
E19C BA0010                     MOV     DX,1000H                ; SEGMENT FOR I/O RAM
E19F 8AE0                       MOV     AH,AL
E1A1 B000                       MOV     AL,0
E1A3                    C20:                                    ; FILL_IO:
E1A3 8EC2                       MOV     ES,DX
E1A5 B90080                     MOV     CX,8000H                ; FILL 32K BYTES
E1A8 2BFF                       SUB     DI,DI
E1AA F3                         REP     STOSB
E1AB AA
E1AC 81C20008                   ADD     DX,800H                 ; NEXT SEGMENT VALUE
E1B0 FECB                       DEC     BL
E1B2 75EF                       JNZ     C20                     ; FILL_IO
                        ;----------------------------------------------------------
                        ;       INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP       :
                        ;----------------------------------------------------------
E1B4                    C21:
E1B4 B013                       MOV     AL,13H                  ; ICW1 - EDGE, SNGL, ICW4
E1B6 E620                       OUT     INTA00,AL
E1B8 B008                       MOV     AL,8                    ; SETUP ICW2 - INT TYPE 8 (8-F)
E1BA E621                       OUT     INTA01,AL
E1BC B009                       MOV     AL,9                    ; SETUP ICW4 - BUFFRD,8086 MODE
E1BE E621                       OUT     INTA01,AL
E1C0 2BC0                       SUB     AX,AX                   ; POINT ES TO BEGIN
E1C2 8EC0                       MOV     ES,AX                   ;  OF R/W STORAGE
                        ;----------------------------------------------------------------------
                        ;       CHECK FOR MANUFACTURING TEST 2 TO LOAD TEST PROGRAMS FROM KEYBOARD.:
                        ;----------------------------------------------------------------------

                        ;----- SETUP STACK SEG AND SP

E1C4 B83000                     MOV     AX,STACK                ; GET STACK VALUE
E1C7 8ED0                       MOV     SS,AX                   ; SET THE STACK UP
E1C9 BC0001                     MOV     SP,OFFSET TOS           ; STACK IS READY TO GO
E1CC 81FD3412                   CMP     BP,1234H                ; RESET_FLAG SET?
E1D0 7425                       JE      C25                     ; YES - SKIP MFG TEST
E1D2 2BFF                       SUB     DI,DI
E1D4 8EDF                       MOV     DS, DI
E1D6 BB2400                     MOV     BX, 24H
E1D9 C70747FF                   MOV     WORD PTR [BX],OFFSET D11 ; SET UP KB INTERRUPT
E1DD 43                         INC     BX
E1DE 43                         INC     BX
E1DF 8C0F                       MOV     [BX],CS
E1E1 E85F04                     CALL    KBD_RESET               ; READ IN KB RESET CODE TO BL
E1E4 80FB65                     CMP     BL,065H                 ; IS THIS MANUFACTURING TEST 2?
E1E7 750E                       JNZ     C25                     ; JUMP IF NOT MAN. TEST
E1E9 B2FF                       MOV     DL,255                  ; READ IN TEST PROGRAM
E1EB                    C22:
E1EB E86204                     CALL    SP_TEST
E1EE 8AC3                       MOV     AL,BL
E1F0 AA                         STOSB
```

Appendix A

```
LOC OBJ                 SOURCE

E1F1 FECA                      DEC     DL
E1F3 75F6                      JNZ     C22                        ; JUMP IF NOT DONE YET
E1F5 CD3E                      INT     3EH                        ; SET INTERRUPT TYPE 62 ADDRESS F8H
E1F7                    C25:

                        ;----- SET UP THE BIOS INTERRUPT VECTORS TO TEMP INTERRUPT

E1F7 B92000                    MOV     CX,32                      ; FILL ALL 32 INTERRUPTS
E1FA 2BFF                      SUB     DI,DI                      ; FIRST INTERRUPT LOCATOIN
E1FC                    D3:
E1FC B847FF                    MOV     AX,OFFSET D11              ; MOVE ADDR OF INTR PROC TO TBL
E1FF AB                        STOSW
E200 8CC8                      MOV     AX,CS                      ; GET ADDR OF INTR PROC SEG
E202 AB                        STOSW
E203 E2F7                      LOOP    D3                         ; VECTBL0

                        ;----- SET UP OTHER INTERRUPTS AS NECESSARY

E205 C7060800C3E2              MOV     NMI_PTR,OFFSET NMI_INT  ; NMI INTERRUPT
E20B C706140054FF              MOV     INT5_PTR,OFFSET PRINT_SCREEN    ; PRINT SCREEN
E211 C706620000F6              MOV     BASIC_PTR+2,0F600H       ; SEGMENT FOR CASSETTE BASIC

                        ;-------------------------------------------------------------
                        ;       8259 INTERRUPT CONTROLLER TEST                       :
                        ; DESCRIPTION                                                :
                        ;       READ/WRITE THE INTERRUPT MASK REGISTER (IMR) WITH ALL  :
                        ;       ONES AND ZEROES. ENABLE SYSTEM INTERRUPTS.  MASK DEVICE :
                        ;       INTERRUPTS OFF. CHECK FOR NOT INTERRUPTS (UNEXPECTED).  :
                        ;-------------------------------------------------------------

                        ;----- TEST THE IMR REGISTER

E217 BA2100                    MOV     DX,0021H                   ; POINT INTR. CHIP ADDR 21
E21A B000                      MOV     AL,0                       ; SET IMR TO ZERO
E21C EE                        OUT     DX,AL
E21D EC                        IN      AL,DX                      ; READ IMR
E21E 0AC0                      OR      AL,AL                      ; IMR = 0?
E220 7515                      JNZ     D6                         ; GO TO ERR ROUTINE IF NOT 0
E222 B0FF                      MOV     AL,0FFH                    ; DISABLE DEVICE INTERRUPTS
E224 EE                        OUT     DX,AL                      ; WRITE TO IMR
E225 EC                        IN      AL,DX                      ; READ IMR
E226 0401                      ADD     AL,1                       ; ALL IMR BIT ON?
E228 750D                      JNZ     D6                         ; NO - GO TO ERR ROUTINE

                        ;----- CHECK FOR NOT INTERRUPTS

                        ;----- INTERRUPTS ARE MASKED OFF.  CHECK THAT NO INTERRUPTS OCCUR.

E22A 32E4                      XOR     AH,AH                      ; CLEAR AH REG
E22C FB                        STI                                ; ENABLE EXTERNAL INTERRUPTS
E22D 2BC9                      SUB     CX,CX                      ; WAIT 1 SEC FOR ANY INTRS THAT
E22F                    D4:
E22F E2FE                      LOOP    D4                         ;  MIGHT OCCUR
E231                    D5:
E231 E2FE                      LOOP    D5
E233 0AE4                      OR      AH,AH                      ; DID ANY INTERRUPTS OCCUR?
E235 7408                      JZ      D7                         ; NO - GO TO NEXT TEST
E237                    D6:
E237 BA0101                    MOV     DX,101H                    ; BEEP SPEAKER IF ERROR
E23A E89203                    CALL    ERR_BEEP                   ; GO TO BEEP SUBROUTINE
E23D FA                        CLI
E23E F4                        HLT                                ; HALT THE SYSTEM
                        ;---------------------------------------------
                        ;       8253 TIMER CHECKOUT                  :
                        ; DESCRIPTION                                :
                        ;       VERIFY THAT THE SYSTEM TIMER (0)     :
                        ;       DOESN'T COUNT TOO FAST OR TOO SLOW.  :
                        ;---------------------------------------------
E23F                    D7:
E23F B0FE                      MOV     AL,0FEH                    ; MASK ALL INTRS EXCEPT LVL 0
E241 EE                        OUT     DX,AL                      ; WRITE THE 8259 IMR
E242 B010                      MOV     AL,00010000B               ; SEL TIM 0, LSB, MODE 0, BINARY
E244 E643                      OUT     TIM_CTL,AL                 ; WRITE TIMER CONTROL MODE REG
E246 B91600                    MOV     CX,16H                     ; SET PGM LOOP CNT
E249 8AC1                      MOV     AL,CL                      ; SET TIMER 0 CNT REG
E24B E640                      OUT     TIMER0,AL                  ; WRITE TIMER 0 CNT REG
```

```
LOC OBJ              LINE    SOURCE

E24D                         D8:
E24D F6C4FF                          TEST    AH,DFFH                 ; DID TIMER 0 INTERRUPT OCCUR?
E250 7504                            JNZ     D9                      ; YES - CHECK TIMER OP FOR SLOW TIME
E252 E2F9                            LOOP    D8                      ; WAIT FOR INTR FOR SPECIFIED TIME
E254 EBE1                            JMP     D6                      ; TIMER 0 INTR DIDN'T OCCUR - ERR
E256                         D9:
E256 B112                            MOV     CL,18                   ; SET PGM LOOP CNT
E258 B0FF                            MOV     AL,0FFH                 ; WRITE TIMER 0 CNT REG
E25A E640                            OUT     TIMERO,AL
E25C B8FE00                          MOV     AX,DFEH
E25F EE                              OUT     DX,AL
E260                         D10:
E260 F6C4FF                          TEST    AH,DFFH                 ; DID TIMER 0 INTERRUPT OCCUR?
E263 75D2                            JNZ     D6                      ; YES - TIMER CNTING TOO FAST, ERR
E265 E2F9                            LOOP    D10                     ; WAIT FOR INTR FOR SPECIFIED TIME

                             ;----- ESTABLISH BIOS SUBROUTINE CALL INTERRUPT VECTORS

E267 1E                              PUSH    DS                      ; SAVE POINTER TO DATA AREA
E268 BF4000                          MOV     DI,OFFSET VIDEO_INT     ; SETUP ADDR TO INTR AREA
E26B 0E                              PUSH    CS
E26C 1F                              POP     DS                      ; SETUP ADDR OF VECTOR TABLE
E26D BE03FF90                        MOV     SI,OFFSET VECTOR_TABLE+16       ; START WITH VIDEO ENTRY
E271 B91000                          MOV     CX,16

                             ;----- SETUP TIMER 0 TO MODE 3

E274 B0FF                            MOV     AL,0FFH                 ; DISABLE ALL DEVICE INTERRUPTS
E276 EE                              OUT     DX,AL
E277 B036                            MOV     AL,36H                  ; SEL TIM 0,LSB,MSB,MODE 3
E279 E643                            OUT     TIMER+3,AL              ; WRITE TIMER MODE REG
E27B B000                            MOV     AL,0
E27D E640                            OUT     TIMER,AL                ; WRITE LSB TO TIMER 0 REG
E27F                         E1A:
E27F A5                              MOVSW                           ; MOVE VECTOR TABLE TO RAM
E280 47                              INC     DI                      ; MOVE PAST SEGMENT POINTER
E281 47                              INC     DI
E282 E2FB                            LOOP    E1A
E284 E640                            OUT     TIMER,AL                ; WRITE MSB TO TIMER 0 REG
E286 1F                              POP     DS                      ; RECOVER DATA SEG POINTER

                             ;----- SETUP TIMER 0 TO BLINK LED IF MANUFACTURING TEST MODE

E287 E8B903                          CALL    KBD_RESET               ; SEND SOFTWARE RESET TO KEYBRD
E28A 80FBAA                          CMP     BL,0AAH                 ; SCAN CODE 'AA' RETURNED?
E28D 741E                            JE      E6                      ; YES - CONTINUE (NON MFG MODE)
E28F B03C                            MOV     AL,3CH                  ; EN KBD, SET KBD CLK LINE LOW
E291 E661                            OUT     PORT_B,AL               ; WRITE 8255 PORT B
E293 90                              NOP
E294 90                              NOP
E295 E460                            IN      AL,PORT_A               ; WAS A BIT CLOCKED IN?
E297 24FF                            AND     AL,0FFH
E299 750E                            JNZ     E2                      ; YES - CONTINUE (NON MFG MODE)
E29B FE061204                        INC     DATA_AREA[OFFSET MFG_TST]       ; ELSE SET SW FOR MFG TEST MODE
E29F C70620006DE6                    MOV     INT_ADDR,OFFSET BLINK_INT       ; SETUP TIMER INTR TO BLINK LED
E2A5 B0FE                            MOV     AL,0FEH                 ; ENABLE TIMER INTERRUPT
E2A7 E621                            OUT     INTA01,AL
E2A9                         E2:                                     ; JUMPER_NOT_IN:
E2A9 B0CC                            MOV     AL,0CCH                 ; RESET THE KEYBOARD
E2AB E661                            OUT     PORT_B,AL

                             ;----------------------------------------------------------
                             ;       INITIALIZE AND START CRT CONTROLLER (6845)        :
                             ;       TEST VIDEO READ/WRITE STORAGE.                    :
                             ; DESCRIPTION                                             :
                             ;       RESET THE VIDEO ENABLE SIGNAL.                    :
                             ;       SELECT ALPHANUMERIC MODE, 40 * 25, B & W.         :
                             ;       READ/WRITE DATA PATTERNS TO STG. CHECK STG        :
                             ;       ADDRESSABILITY.                                   :
                             ;----------------------------------------------------------
E2AD                         E6:
E2AD E460                            IN      AL,PORT_A               ; READ SENSE SWITCHES
E2AF B400                            MOV     AH,0
E2B1 A31004                          MOV     DATA_WORD[OFFSET EQUIP_FLAG],AX ; STORE SENSE SW INFO
E2B4                         E6A:
E2B4 2430                            AND     AL,30H                  ; ISOLATE VIDEO SWS
E2B6 7529                            JNZ     E7                      ; VIDEO SWS SET TO 0?
```

Appendix A

```
LOC OBJ                  SOURCE

E2B8 C706400053FF                MOV    VIDEO_INT,OFFSET DUMMY_RETURN
E2BE E9A200                      JMP    E18_1                    ; SKIP VIDEO TESTS FOR BURN-IN

E2C3                             ORG    0E2C3H
E2C3                     NMI_INT PROC   NEAR
E2C3 50                          PUSH   AX                       ; SAVE ORIG CONTENTS OF AX
E2C4 E462                        IN     AL,PORT_C
E2C6 A8C0                        TEST   AL,0C0H                  ; PARITY CHECK?
E2C8 7415                        JZ     D14                      ; NO, EXIT FROM ROUTINE
E2CA BEDAFF90                    MOV    SI,OFFSET D1             ; ADDR OF ERROR MSG
E2CE A840                        TEST   AL,40H                   ; I/O PARITY CHECK
E2D0 7504                        JNZ    D13                      ; DISPLAY ERROR MSG
E2D2 BE23FF90                    MOV    SI,OFFSET D2             ; MUST BE PLANAR
E2D6                     D13:
E2D6 2BC0                        SUB    AX,AX                    ; INIT AND SET MODE FOR VIDEO
E2D8 CD10                        INT    10H                      ; CALL VIDEO_IO PROCEDURE
E2DA E80003                      CALL   P_MSG                    ; PRINT ERROR MSG
E2DD FA                          CLI
E2DE F4                          HLT                             ; HALT SYSTEM
E2DF                     D14:
E2DF 58                          POP    AX                       ; RESTORE ORIG CONTENTS OF AX
E2E0 CF                          IRET
                         NMI_INT ENDP
E2E1                     E7:                                     ; TEST_VIDEO:
E2E1 3C30                        CMP    AL,30H                   ; B/W CARD ATTACHED?
E2E3 7408                        JE     E8                       ; YES - SET MODE FOR B/W CARD
E2E5 FEC4                        INC    AH                       ; SET COLOR MODE FOR COLOR CD
E2E7 3C20                        CMP    AL,20H                   ; 80X25 MODE SELECTED?
E2E9 7502                        JNE    E8                       ; NO - SET MODE FOR 40X25
E2EB B403                        MOV    AH,3                     ; SET MODE FOR 80X25
E2ED                     E8:
E2ED 86E0                        XCHG   AH,AL                    ; SET_MODE
E2EF 50                          PUSH   AX                       ; SAVE VIDEO MODE ON STACK
E2F0 2AE4                        SUB    AH,AH                    ; INITIALIZE TO ALPHANUMERIC MD
E2F2 CD10                        INT    10H                      ; CALL VIDEO_IO
E2F4 58                          POP    AX                       ; RESTORE VIDEO SENSE SWS IN AH
E2F5 50                          PUSH   AX                       ; RESAVE VALUE
E2F6 BB00B0                      MOV    BX,0B000H                ; BEG VIDEO RAM ADDR B/W CD
E2F9 BAB803                      MOV    DX,3B8H                  ; MODE REG FOR B/W
E2FC B90010                      MOV    CX,4096                  ; RAM BYTE CNT FOR B/W CD
E2FF B001                        MOV    AL,1                     ; SET MODE FOR BW CARD
E301 80FC30                      CMP    AH,30H                   ; B/W VIDEO CARD ATTACHED?
E304 7408                        JE     E9                       ; YES - GO TEST VIDEO STG
E306 B7B8                        MOV    BH,0B8H                  ; BEG VIDEO RAM ADDR COLOR CD
E308 B2D8                        MOV    DL,0D8H                  ; MODE REG FOR COLOR CD
E30A B540                        MOV    CH,40H                   ; RAM BYTE CNT FOR COLOR CD
E30C FEC8                        DEC    AL                       ; SET MODE TO 0 FOR COLOR CD
E30E                     E9:                                     ; TEST_VIDEO_STG:
E30E EE                          OUT    DX,AL                    ; DISABLE VIDEO FOR COLOR CD
E30F 81FD3412                    CMP    BP,1234H                 ; POD INITIATED BY KBD RESET?
E313 8EC3                        MOV    ES,BX                    ; POINT ES TO VIDEO RAM STG
E315 7407                        JE     E10                      ; YES - SKIP VIDEO RAM TEST
E317 8EDB                        MOV    DS,BX                    ; POINT DS TO VIDEO RAM STG
                                 ASSUME DS:NOTHING,ES:NOTHING
E319 E8FFFC                      CALL   STGTST_CNT               ; GO TEST VIDEO R/W STG
E31C 7532                        JNE    E17                      ; R/W STG FAILURE - BEEP SPK
                         ;-----------------------------------------------------------
                         ;       SETUP VIDEO DATA ON SCREEN FOR VIDEO LINE TEST.   :
                         ; DESCRIPTION                                             :
                         ;       ENABLE VIDEO SIGNAL AND SET MODE.                 :
                         ;       DISPLAY A HORIZONTAL BAR ON SCREEN.               :
                         ;-----------------------------------------------------------
E31E                     E10:
E31E 58                          POP    AX                       ; GET VIDEO SENSE SWS (AH)
E31F 50                          PUSH   AX                       ; SAVE IT
E320 B400                        MOV    AH,0                     ; ENABLE VIDEO AND SET MODE
E322 CD10                        INT    10H                      ; VIDEO
E324 B82070                      MOV    AX,7020H                 ; WRT BLANKS IN REVERSE VIDEO
E327 2BFF                        SUB    DI,DI                    ; SETUP STARTING LOC
E329 B92800                      MOV    CX,40                    ; NO. OF BLANKS TO DISPLAY
E32C F3                          REP    STOSW                    ; WRITE VIDEO STORAGE
E32D AB
                         ;----------------------------------------------------
                         ;       CRT INTERFACE LINES TEST                    :
                         ; DESCRIPTION                                       :
                         ;       SENSE ON/OFF TRANSITION OF THE VIDEO ENABLE :
```

```
LOC OBJ                 SOURCE

                        ;       AND HORIZONTAL SYNC LINES.                        :
                        ;----------------------------------------------------------
E32E 58                         POP     AX              ; GET VIDEO SENSE SW INFO
E32F 50                         PUSH    AX              ; SAVE IT
E330 80FC30                     CMP     AH,30H          ; B/W CARD ATTACHED?
E333 BABA03                     MOV     DX,03BAH        ; SETUP ADDR OF BW STATUS PORT
E336 7402                       JE      E11             ; YES - GO TEST LINES
E338 B2DA                       MOV     DL,0DAH         ; COLOR CARD IS ATTACHED
E33A                    E11:                            ; LINE_TST:
E33A B408                       MOV     AH,8
E33C                    E12:                            ; OFLOOP_CNT:
E33C 2BC9                       SUB     CX,CX
E33E                    E13:
E33E EC                         IN      AL,DX           ; READ CRT STATUS PORT
E33F 22C4                       AND     AL,AH           ; CHECK VIDEO/HORZ LINE
E341 7504                       JNZ     E14             ; ITS ON - CHECK IF IT GOES OFF
E343 E2F9                       LOOP    E13             ; LOOP TILL ON OR TIMEOUT
E345 EB09                       JMP     SHORT E17       ; GO PRINT ERROR MSG
E347                    E14:
E347 2BC9                       SUB     CX,CX
E349                    E15:
E349 EC                         IN      AL,DX           ; READ CRT STATUS PORT
E34A 22C4                       AND     AL,AH           ; CHECK VIDEO/HORZ LINE
E34C 740A                       JZ      E16             ; ITS ON - CHECK NEXT LINE
E34E E2F9                       LOOP    E15             ; LOOP IF OFF TILL IT GOES ON
E350                    E17:                            ;  CRT_ERR
E350 BA0201                     MOV     DX,102H
E353 E87902                     CALL    ERR_BEEP        ; GO BEEP SPEAKER
E356 EB06                       JMP     SHORT E18
E358                    E16:                            ; NXT_LINE
E358 B103                       MOV     CL,3            ; GET NEXT BIT TO CHECK
E35A D2EC                       SHR     AH,CL
E35C 75DE                       JNZ     E12             ; GO CHECK HORIZONTAL LINE
E35E                    E18:                            ;  DISPLAY_CURSOR:
E35E 58                         POP     AX              ; GET VIDEO SENSE SWS (AH)
E35F B400                       MOV     AH,0            ; SET MODE AND DISPLAY CURSOR
E361 CD10                       INT     10H             ; CALL VIDEO I/O PROCEDURE

E363                    E18_1:
E363 BA00C0                     MOV     DX,0C000H
E366                    E18A:
E366 8EDA                       MOV     DS,DX
E368 2BDB                       SUB     BX,BX
E36A 8B07                       MOV     AX,[BX]         ; GET FIRST 2 LOCATIONS
E36C 53                         PUSH    BX
E36D 5B                         POP     BX              ; LET BUS SETTLE
E36E 3D55AA                     CMP     AX,0AA55H       ; PRESENT?
E371 7505                       JNZ     E18B            ; NO? GO LOOK FOR OTHER MODULES
E373 E80E03                     CALL    ROM_CHECK       ; GO SCAN MODULE
E376 EB04                       JMP     SHORT E18C
E378                    E18B:
E378 81C28000                   ADD     DX,0080H        ; POINT TO NEXT 2K BLOCK
E37C                    E18C:
E37C 81FA00C8                   CMP     DX,0C800H       ; TOP OF VIDEO ROM AREA YET?
E380 7CE4                       JL      E18A            ; GO SCAN FOR ANOTHER MODULE
                        ;------------------------------------------------------------
                        ; EXPANSION I/O BOX TEST                                     :
                        ;       CHECK TO SEE IF EXPANSION BOX PRESENT - IF INSTALLED, :
                        ;       TEST DATA AND ADDRESS BUSES TO I/O BOX.              :
                        ; ERROR='1801'                                               :
                        ;------------------------------------------------------------

                        ;----- DETERMINE IF BOX IS PRESENT

E382                    EXP_IO:                         ; (CARD WAS ENABLED EARLIER)
E382 BA1002                     MOV     DX,0210H        ; CONTROL PORT ADDRESS
E385 B85555                     MOV     AX,5555H        ; SET DATA PATTERN
E388 EE                         OUT     DX,AL
E389 B001                       MOV     AL,01H
E38B EC                         IN      AL,DX           ; RECOVER DATA
E38C 3AC4                       CMP     AL,AH           ; REPLY?
E38E 7534                       JNE     E19             ; NO RESPONSE, GO TO NEXT TEST
E390 F7D0                       NOT     AX              ; MAKE DATA=AAAA
E392 EE                         OUT     DX,AL
E393 B001                       MOV     AL,01H
E395 EC                         IN      AL,DX           ; RECOVER DATA
E396 3AC4                       CMP     AL,AH
```

Appendix A

```
LOC OBJ              SOURCE

E398 752A                    JNE     E19                     ; NO ANSWER=NEXT TEST

                     ;----- CHECK ADDRESS AND DATA BUS

E39A                 EXP1:
E39A 8BD8                    MOV     BX,AX
E39C BA1402                  MOV     DX,0214H                ; LOAD DATA  REG ADDRESS
E39F 2E8807                  MOV     CS:[BX],AL              ; WRITE ADDRESS F0000+BX
E3A2 EE                      OUT     DX,AL                   ; WRITE DATA
E3A3 90                      NOP
E3A4 EC                      IN      AL,DX                   ; READ DATA
E3A5 3AC7                    CMP     AL,BH
E3A7 7514                    JNE     EXP_ERR
E3A9 42                      INC     DX                      ; DX=215H (ADDR. HI REG)
E3AA EC                      IN      AL,DX
E3AB 3AC4                    CMP     AL,AH                   ; COMPARE TO HI ADDRESS
E3AD 750E                    JNE     EXP_ERR
E3AF 42                      INC     DX                      ; DX-216H (ADDR. LOW REG)
E3B0 EC                      IN      AL,DX
E3B1 3AC4                    CMP     AL,AH                   ; ADDR. LOW OK?
E3B3 7508                    JNE     EXP_ERR
E3B5 F7D0                    NOT     AX                      ; INVERT AX
E3B7 3CAA                    CMP     AL,0AAH                 ; BACK TO STARTING VALUE (AAAA) YET
E3B9 7409                    JE      E19                     ; GO ON TO NEXT TEST IF SO
E3BB EBDD                    JMP     EXP1                    ; LOOP BACK THROUGH WITH DATA OF 5555
E3BD                 EXP_ERR:
E3BD BEE0FE90                MOV     SI,OFFSET F3B
E3C1 E8F602                  CALL    P_MSG
                     ;----------------------------------------------------------------
                     ;       ADDITIONAL READ/WRITE STORAGE TEST                      :
                     ; DESCRIPTION                                                   :
                     ;       WRITE/READ DATA PATTERNS TO ANY READ/WRITE STORAGE      :
                     ;       AFTER THE BASIC 16K. STORAGE ADDRESSABILITY IS CHECKED. :
                     ;----------------------------------------------------------------
                             ASSUME  DS:DATA
E3C4                 E19:

                     ;----- DETERMINE RAM SIZE ON PLANAR BOARD

E3C4 E8771B                  CALL    DDS
E3C7 A01000                  MOV     AL,BYTE PTR EQUIP_FLAG  ; GET SENSE SWS INFO
E3CA 240C                    AND     AL,0CH                  ; ISOLATE RAM SIZE SWS
E3CC B404                    MOV     AH,4
E3CE F6E4                    MUL     AH
E3D0 0410                    ADD     AL,16                   ; ADD BASIC 16K
E3D2 8BD0                    MOV     DX,AX                   ; SAVE PLANAR RAM SIZE IN DX
E3D4 8BD8                    MOV     BX,AX                   ;  AND IN BX

                     ;----- DETERMINE IO CHANNEL RAM SIZE

E3D6 A11500                  MOV     AX,IO_RAM_SIZE          ; GET IO CHANNEL RAM SIZE
E3D9 83FB40                  CMP     BX,40H                  ; PLANAR RAM SIZE = 64K?
E3DC 7402                    JE      E20                     ; YES - ADD IO CHN RAM SIZE
E3DE 2BC0                    SUB     AX,AX                   ; NO - DON'T ADD ANY IO RAM
E3E0                 E20:                                    ; ADD_IO_SIZE:
E3E0 03C3                    ADD     AX,BX                   ; SUM TOTAL RAM SIZE
E3E2 A31300                  MOV     MEMORY_SIZE,AX          ; SETUP MEMORY SIZE PARM
E3E5 81FD3412                CMP     BP,1234H                ; POD INITIATED BY KBD RESET?
E3E9 1E                      PUSH    DS                      ; SAVE DATA SEGMENT
E3EA 744F                    JE      TST12                   ; YES - SKIP MEMORY TEST

                     ;----- TEST ANY OTHER READ/WRITE STORAGE AVAILABLE

E3EC BB0004                  MOV     BX,400H
E3EF B91000                  MOV     CX,16
E3F2                 E21:
E3F2 3BD1                    CMP     DX,CX                   ; ANY MORE STG TO BE TESTED?
E3F4 762D                    JBE     E23                     ; NO - GO TO NEXT TEST
E3F6 8EDB                    MOV     DS,BX                   ; SETUP STG ADDR IN DS AND ES
E3F8 8EC3                    MOV     ES,BX
E3FA 83C110                  ADD     CX,16                   ; INCREMENT STG BYTE COUNTER
E3FD 81C30004                ADD     BX,400H                 ; SET POINTER TO NEXT 16K BLK
E401 51                      PUSH    CX                      ; SAVE REGS
E402 53                      PUSH    BX
E403 52                      PUSH    DX
E404 E811FC                  CALL    STGTST                  ; GO TEST A 16K BLK OF STG
E407 5A                      POP     DX
```

```
LOC OBJ                 LINE    SOURCE

E408 5B                          POP     BX                      ; RESTORE REGS
E409 59                          POP     CX
E40A 74E6                        JE      E21                     ; CHECK IF MORE STG TO TEST

                        ;----- PRINT FAILING ADDRESS AND XOR'ED PATTERN IF OATA COMPARE ERROR

E40C 8CDA                        MOV     DX,OS                   ; CONVERT FAILING HIGH-ORDER
E40E 8AE8                        MOV     CH,AL                   ; SAVE FAILING BIT PATTERN
E410 8AC6                        MOV     AL,OH                   ; GET FAILING AOOR
E412 E81002                      CALL    XPC_BYTE                ; CONVERT AND PRINT CODE
E415 8AC5                        MOV     AL,CH                   ; GET FAILING BIT PATTERN
E417 E80B02                      CALL    XPC_BYTE                ; CONVERT AND PRINT CODE
E41A BE67FA90                    MOV     SI,OFFSET E1            ; SETUP ADDRESS OF ERROR MSG
E41E E89902                      CALL    P_MSG                   ; PRINT ERROR MSG
E421                    E22:
E421 EB18                        JMP     SHORT TST12             ; GO TO NEXT TEST
E423                    E23:                                     ; STG_TEST_OONE
E423 1F                          POP     DS                      ; POINT OS TO DATA SEGMENT
E424 1E                          PUSH    DS
E425 8B161500                    MOV     OX,IO_RAM_SIZE          ; GET IO CHANNEL RAM SIZE
E429 0BD2                        OR      OX,OX                   ; SET FLAG RESULT
E42B 740E                        JZ      TST12                   ; NO IO RAM, GO TO NEXT TEST
E42D B90000                      MOV     CX,0
E430 81FB0010                    CMP     BX,1000H                ; HAS IO RAM BEEN TESTEO
E434 7705                        JA      TST12                   ; YES - GO TO NEXT TEST
E436 BB0010                      MOV     BX,1000H                ; SETUP BEG LOC FOR IO RAM
E439 EBB7                        JMP     E21                     ; GO TEST IO CHANNEL RAM
                        ;------------------------------------------------------------
                        ;       KEYBOARO TEST                                       :
                        ; OESCRIPTION                                               :
                        ;       RESET THE KEYBOARO AND CHECK THAT SCAN COOE         :
                        ;       'AA' IS RETURNED TO THE CPU.  CHECK FOR STUCK       :
                        ;        KEYS.                                              :
                        ;------------------------------------------------------------
                                 ASSUME  OS:OATA
E43B                    TST12:
E43B 1F                          POP     OS
E43C 803E120001                  CMP     MFG_TST,1               ; MANUFACTURING TEST MODE?
E441 742A                        JE      F7                      ; YES - SKIP KEYBOARO TEST
E443 E8F001                      CALL    KBO_RESET               ; ISSUE SOFTWARE RESET TO KEYBRO
E446 E31E                        JCXZ    F6                      ; PRINT ERR MSG IF NO INTERRUPT
E448 B040                        MOV     AL,40H                  ; ENABLE KEYBOARO
E44A E661                        OUT     PORT_B,AL
E44C 80FBAA                      CMP     BL,0AAH                 ; SCAN CODE AS EXPECTEO?
E44F 7515                        JNE     F6                      ; NO - DISPLAY ERROR MSG

                        ;----- CHECK FOR STUCK KEYS

E451 B0CC                        MOV     AL,0CCH                 ; CLR KBO, SET CLK LINE HIGH
E453 E661                        OUT     PORT_B,AL
E455 B04C                        MOV     AL,4CH                  ; ENABLE KBD,CLK IN NEXT BYTE
E457 E661                        OUT     PORT_B,AL
E459 2BC9                        SUB     CX,CX
E45B                    F5:                                      ; KBO_WAIT
E45B E2FE                        LOOP    F5                      ; OELAY FOR A WHILE
E45D E460                        IN      AL,KBO_IN               ; CHECK FOR STUCK KEYS
E45F 3C00                        CMP     AL,0                    ; SCAN COOE = 0?
E461 740A                        JE      F7                      ; YES - CONTINUE TESTING
E463 E8BF01                      CALL    XPC_BYTE                ; CONVERT AND PRINT
E466 BE33FF90           F6:      MOV     SI,OFFSET F1            ; GET MSG AOOR
E46A E84002                      CALL    P_MSG                   ; PRINT MSG ON SCREEN

                        ;----- SETUP INTERRUPT VECTOR TABLE

E46D                    F7:                                      ; SETUP_INT_TABLE:
E46D 2BC0                        SUB     AX,AX
E46F 8EC0                        MOV     ES,AX
E471 B90800                      MOV     CX,8                    ; GET VECTOR CNT
E474 1E                          PUSH    DS                      ; SAVE DATA SEGMENT
E475 0E                          PUSH    CS                      ; SETUP DS SEG REG
E476 1F                          POP     DS
E477 BEF3FE90                    MOV     SI,OFFSET VECTOR_TABLE
E47B BF2000                      MOV     DI,OFFSET INT_PTR
E47E                    F7A:
E47E A5                          MOVSW
E47F 47                          INC     DI                      ; SKIP OVER SEGMENT
E480 47                          INC     DI
```

Appendix A

```
LOC OBJ                 LINE     SOURCE

E481 E2FB                                LOOP    F7A
                                ;-------------------------------------------------------
                                ;        CASSETTE DATA WRAP TEST                       :
                                ; DESCRIPTION                                          :
                                ;        TURN CASSETTE MOTOR OFF. WRITE A BIT OUT TO THE :
                                ;        CASSETTE DATA BUS. VERIFY THAT CASSETTE DATA   :
                                ;        READ IS WITHIN A VALID RANGE.                  :
                                ;-------------------------------------------------------

                                ;----- TURN THE CASSETTE MOTOR OFF

E483                            TST13:
E483 1F                                 POP     DS
E484 1E                                 PUSH    DS
E485 B04D                               MOV     AL,04DH                 ; SET TIMER 2 SPK OUT, AND CASST
E487 E661                               OUT     PORT_B,AL               ; OUT BITS ON, CASSETTE MOT OFF

                                ;----- WRITE A BIT

E489 B0FF                               MOV     AL,0FFH                 ; DISABLE TIMER INTERRUPTS
E48B E621                               OUT     INTA01,AL
E48D B0B6                               MOV     AL,0B6H                 ; SEL TIM 2, LSB, MSB, MD 3
E48F E643                               OUT     TIMER+3,AL              ; WRITE 8253 CMD/MODE REG
E491 B8D304                             MOV     AX,1235                 ; SET TIMER 2 CNT FOR 1000 USEC
E494 E642                               OUT     TIMER+2,AL              ; WRITE TIMER 2 COUNTER REG
E496 8AC4                               MOV     AL,AH                   ; WRITE MSB
E498 E642                               OUT     TIMER+2,AL

                                ;----- READ CASSETTE INPUT

E49A E462                               IN      AL,PORT_C               ; READ VALUE OF CASS IN BIT
E49C 2410                               AND     AL,10H                  ; ISOLATE FROM OTHER BITS
E49E A26B00                             MOV     LAST_VAL,AL
E4A1 E8D514                             CALL    READ_HALF_BIT
E4A4 E8D214                             CALL    READ_HALF_BIT
E4A7 E30C                               JCXZ    F8                      ; CAS_ERR
E4A9 81FB4005                           CMP     BX,MAX_PERIOD
E4AD 7306                               JNC     F8                      ; CAS_ERR
E4AF 81FB1004                           CMP     BX,MIN_PERIOD
E4B3 7307                               JNC     ROM_SCAN                ; GO TO NEXT TEST IF OK
E4B5                            F8:                                     ; CAS_ERR
E4B5 BE39FF90                           MOV     SI,OFFSET F2            ; CASSETTE WRAP FAILED
E4B9 E8FE01                             CALL    P_MSG                   ; GO PRINT ERROR MSG
                                ;------------------------------------------------------------------------
                                ;       CHECK FOR OPTIONAL ROM FROM C800O->F4000 IN 2K INCREMENTS      :
                                ;       (A VALID MODULE HAS '55AA' IN THE FIRST 2 LOCATIONS, LENGTH     :
                                ;       INDICATOR (LENGTH/512) IN THE 3RD LOCATION AND TEST/INIT.       :
                                ;       CODE STARTING IN THE 4TH LOCATION.)                             :
                                ;------------------------------------------------------------------------
E4BC                            ROM_SCAN:
E4BC BA00C8                             MOV     DX,0C800H               ; SET BEGINNING ADDRESS
E4BF                            ROM_SCAN_1:
E4BF 8EDA                               MOV     DS,DX
E4C1 2BDB                               SUB     BX,BX                   ; SET BX=0000
E4C3 8B07                               MOV     AX,[BX]                 ; GET 1ST WORD FROM MODULE
E4C5 3D55AA                             CMP     AX,0AA55H               ; = TO ID WORD?
E4C8 7505                               JNZ     NEXT_ROM                ; PROCEED TO NEXT ROM IF NOT
E4CA E88701                             CALL    ROM_CHECK               ; GO DO CHECKSUM AND CALL
E4CD EB04                               JMP     SHORT ARE_WE_DONE       ; CHECK FOR END OF ROM SPACE
E4CF                            NEXT_ROM:
E4CF 81C28000                           ADD     DX,0080H                ; POINT TO NEXT 2K ADDRESS
E4D3                            ARE_WE_DONE:
E4D3 81FA00F6                           CMP     DX,0F600H               ; AT F6000 YET?
E4D7 7CE6                               JL      ROM_SCAN_1              ; GO CHECK ANOTHER ADD. IF NOT
E4D9 EB0190                             JMP     BASE_ROM_CHK            ; GO CHECK BASIC ROM
                                ;----------------------------------------------
                                ;        ROS CHECKSUM II                      :
                                ; DESCRIPTION                                 :
                                ;        A CHECKSUM IS DONE FOR THE 4 ROS     :
                                ;        MODULES CONTAINING BASIC CODE        :
                                ;----------------------------------------------
E4DC                            BASE_ROM_CHK:
E4DC                            E4:
E4DC 2BDB                               SUB     BX,BX                   ; SETUP STARTING ROS ADDR
E4DE 8EDA                               MOV     DS,DX
E4E0 E86907                             CALL    ROS_CHECKSUM            ; CHECK ROS
```

```
LOC OBJ                 SOURCE

E4E3 7403                       JE      E5                      ; CONTINUE IF OK
E4E5 E82103                     CALL    ROM_ERR                 ; POST ERROR
E4E8                    E5:
E4E8 80C602                     ADD     DH,02H                  ; POINT TO NEXT 8K MODULE
E4EB 80FEFE                     CMP     DH,0FEH
E4EE 75EC                       JNZ     E4                      ; YES - CONTINUE
E4F0 1F                         POP     DS                      ; RECOVER DATA SEG PTR
                        ;----------------------------------------------------------------
                        ;       DISKETTE ATTACHMENT TEST                                :
                        ; DESCRIPTION                                                   :
                        ;       CHECK IF IPL DISKETTE DRIVE IS ATTACHED TO SYSTEM. IF ATTACHED, :
                        ;       VERIFY STATUS OF NEC FDC AFTER A RESET. ISSUE A RECAL AND SEEK  :
                        ;       CMD TO FDC AND CHECK STATUS. COMPLETE SYSTEM INITIALIZATION     :
                        ;       THEN PASS CONTROL TO THE BOOT LOADER PROGRAM.                   :
                        ;----------------------------------------------------------------
E4F1                    F9:
E4F1 A01000                     MOV     AL,BYTE PTR EQUIP_FLAG  ; GET SENSE SWS INFO
E4F4 A801                       TEST    AL,01H                  ; IPL DISKETTE DRIVE ATTCH?
E4F6 750A                       JNZ     F10                     ; NO -SKIP THIS TEST
E4F8 803E120001                 CMP     MFG_TST,1               ; MANUFACTURING TEST MODE?
E4FD 753D                       JNE     F15A                    ; NO - GO TO BOOT LOADER
E4FF E959FB                     JMP     START                   ; YES - LOOP POWER-ON-DIAGS
E502                    F10:
E502 E421                       IN      AL,INTA01               ; DISK_TEST
E504 24BF                       AND     AL,0BFH                 ; ENABLE DISKETTE INTERRUPTS
E506 E621                       OUT     INTA01,AL
E508 B400                       MOV     AH,0                    ; RESET NEC FDC
E50A 8AD4                       MOV     DL,AH                   ; (POINT TO DISKETTE)
E50C CD13                       INT     13H                     ; VERIFY STATUS AFTER RESET
E50E 7221                       JC      F13

                        ;----- TURN DRIVE 0 MOTOR ON

E510 BAF203                     MOV     DX,03F2H                ; GET ADDR OF FDC CARD
E513 52                         PUSH    DX                      ; SAVE IT
E514 B01C                       MOV     AL,1CH                  ; TURN MOTOR ON, EN DMA/INT
E516 EE                         OUT     DX,AL                   ; WRITE FDC CONTROL REG
E517 2BC9                       SUB     CX,CX
E519                    F11:                                    ; MOTOR_WAIT:
E519 E2FE                       LOOP    F11                     ; WAIT FOR 1 SECOND
E51B                    F12:                                    ; MOTOR_WAIT1:
E51B E2FE                       LOOP    F12
E51D 33D2                       XOR     DX,DX                   ; SELECT DRIVE 0
E51F B501                       MOV     CH,1                    ; SELECT TRACK 1
E521 88163E00                   MOV     SEEK_STATUS,DL
E525 E85509                     CALL    SEEK                    ; RECALIBRATE DISKETTE
E528 7207                       JC      F13                     ; GO TO ERR SUBROUTINE IF ERR
E52A B522                       MOV     CH,34                   ; SELECT TRACK 34
E52C E84E09                     CALL    SEEK                    ; SEEK TO TRACK 34
E52F 7307                       JNC     F14                     ; OK, TURN MOTOR OFF
E531                    F13:                                    ; DSK_ERR:
E531 BEEAFF90                   MOV     SI,OFFSET F3            ; GET ADDR OF MSG
E535 E88201                     CALL    P_MSG                   ; GO PRINT ERROR MSG

                        ;----- TURN DRIVE 0 MOTOR OFF

E538                    F14:                                    ; DR0_OFF:
E538 B00C                       MOV     AL,0CH                  ; TURN DRIVE 0 MOTOR OFF
E53A 5A                         POP     DX                      ; RECOVER FDC CTL ADDRESS
E53B EE                         OUT     DX,AL

                        ;----- SETUP PRINTER AND RS232 BASE ADDRESSES IF DEVICE ATTACHED

E53C                    F15A:
E53C BE1E00                     MOV     SI,OFFSET KB_BUFFER
E53F 89361A00                   MOV     BUFFER_HEAD,SI          ; SETUP KEYBOARD PARAMETERS
E543 89361C00                   MOV     BUFFER_TAIL,SI
E547 89368000                   MOV     BUFFER_START,SI         ; DEFAULT TO STANDARD BUFFER
E54B 83C620                     ADD     SI,32                   ; (32 BYTES LONG)
E54E 89368200                   MOV     BUFFER_END,SI
E552 E421                       IN      AL,INTA01
E554 24FC                       AND     AL,0FCH                 ; ENABLE TIMER AND KBD INTS
E556 E621                       OUT     INTA01,AL
E558 BD3DE690                   MOV     BP,OFFSET F4            ; PRT_SRC_TBL
E55C 2BF6                       SUB     SI,SI
E55E                    F16:                                    ; PRT_BASE:
E55E 2E8B5600                   MOV     DX,CS:[BP]              ; GET PRINTER BASE ADDR
```

Appendix A

```
LOC OBJ            LINE     SOURCE

E562 B0AA                       MOV     AL,0AAH                 ; WRITE DATA TO PORT A
E564 EE                         OUT     DX,AL
E565 52                         PUSH    DX
E566 EC                         IN      AL,DX                   ; READ PORT A
E567 5A                         POP     DX
E568 3CAA                       CMP     AL,0AAH                 ; DATA PATTERN SAME
E56A 7505                       JNE     F17                     ; NO - CHECK NEXT PRT CD
E56C 895408                     MOV     PRINTER_BASE[SI],DX     ; YES - STORE PRT BASE ADDR
E56F 46                         INC     SI                      ; INCREMENT TO NEXT WORD
E570 46                         INC     SI
E571                        F17:                                ; NO_STORE:
E571 45                         INC     BP                      ; POINT TO NEXT BASE ADDR
E572 45                         INC     BP
E573 81FD43E6                   CMP     BP,OFFSET F4E           ; ALL POSSIBLE ADDRS CHECKED?
E577 75E5                       JNE     F16                     ; PRT_BASE
E579 2BDB                       SUB     BX,BX                   ; POINTER TO RS232 TABLE
E57B BAFA03                     MOV     DX,3FAH                 ; CHECK IF RS232 CD 1 ATTCH?
E57E EC                         IN      AL,DX                   ; READ INTR ID REG
E57F A8F8                       TEST    AL,0F8H
E581 7506                       JNZ     F18
E583 C707F803                   MOV     RS232_BASE[BX],3F8H     ; SETUP RS232 CD #1 ADDR
E587 43                         INC     BX
E588 43                         INC     BX
E589                        F18:
E589 B602                       MOV     DH,02H                  ; CHECK IF RS232 CD 2 ATTCH (AT 2FA)
E58B EC                         IN      AL,DX                   ; READ INTERRUPT ID REG
E58C A8F8                       TEST    AL,0F8H
E58E 7506                       JNZ     F19                     ; BASE_END
E590 C707F802                   MOV     RS232_BASE[BX],2F8H     ; SETUP RS232 CD #2
E594 43                         INC     BX
E595 43                         INC     BX

                            ;----- SET UP EQUIP FLAG TO INDICATE NUMBER OF PRINTERS AND RS232 CARDS

E596                        F19:                                ; BASE_END:
E596 8BC6                       MOV     AX,SI                   ; SI HAS 2* NUMBER OF RS232
E598 B103                       MOV     CL,3                    ; SHIFT COUNT
E59A D2C8                       ROR     AL,CL                   ; ROTATE RIGHT 3 POSITIONS
E59C 0AC3                       OR      AL,BL                   ; OR IN THE PRINTER COUNT
E59E A21100                     MOV     BYTE PTR EQUIP_FLAG+1,AL ; STORE AS SECOND BYTE
E5A1 B201                       MOV     DL,01H                  ; DX=201
E5A3 EC                         IN      AL,DX
E5A4 A80F                       TEST    AL,0FH
E5A6 7505                       JNZ     F20                     ; NO_GAME_CARD
E5A8 800E110010                 OR      BYTE PTR EQUIP_FLAG+1,16
E5AD                        F20:

                            ;----- SET DEFAULT TIMEOUT VALUES FOR PRINTER AND RS232

E5AD 1E                         PUSH    DS
E5AE 07                         POP     ES
E5AF BF7800                     MOV     DI,OFFSET PRINT_TIM_OUT
E5B2 B81414                     MOV     AX,1414H                ; PRINTER DEFAULTS (COUNT=20)
E5B5 AB                         STOSW
E5B6 AB                         STOSW
E5B7 B80101                     MOV     AX,0101H                ; RS232 DEFAULTS=01
E5BA AB                         STOSW
E5BB AB                         STOSW

                            ;----- ENABLE NMI INTERRUPTS

E5BC B080                       MOV     AL,80H                  ; ENABLE NMI INTERRUPTS
E5BE E6A0                       OUT     0A0H,AL
E5C0 803E120001                 CMP     MFG_TST,1               ; MFG MODE?
E5C5 7406                       JE      F21                     ; LOAD_BOOT_STRAP
E5C7 BA0100                     MOV     DX,1
E5CA E80200                     CALL    ERR_BEEP                ; BEEP 1 SHORT TONE

E5CD                        F21:                                ; LOAD_BOOT_STRAP:
E5CD CD19                       INT     19H                     ; BOOTSTRAP

                            ;------------------------------------------------------
                            ;       INITIAL RELIABILITY TEST -- SUBROUTINES        :
                            ;------------------------------------------------------
                                ASSUME  CS:CODE,DS:DATA
                            ;-----------------------------------------------------------------------
                            ; SUBROUTINES FOR POWER ON DIAGNOSTICS                                 :
```

```
LOC OBJ                SOURCE

                       ;        THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SECS) AND ONE OR   :
                       ;        MORE SHORT TONES (1 SEC) TO INDICATE A FAILURE ON THE PLANAR   :
                       ;        BOARD, A BAD RAM MODULE, OR A PROBLEM WITH THE CRT.            :
                       ; ENTRY PARAMETERS:                                                     :
                       ;        DH = NUMBER OF LONG TONES TO BEEP                              :
                       ;        DL = NUMBER OF SHORT TONES TO BEEP                             :
                       ;------------------------------------------------------------------------
E5CF                   ERR_BEEP  PROC  NEAR
E5CF 9C                         PUSHF                           ; SAVE FLAGS
E5D0 FA                         CLI                             ; DISABLE SYSTEM INTERRUPTS
E5D1 1E                         PUSH    DS                      ; SAVE DS REG CONTENTS
E5D2 E86919                     CALL    DDS
E5D5 0AF6                       OR      DH,DH                   ; ANY LONG ONES TO BEEP
E5D7 7418                       JZ      G3                      ; NO, DO THE SHORT ONES
E5D9                   G1:                                      ; LONG_BEEP:
E5D9 B306                       MOV     BL,6                    ; COUNTER FOR BEEPS
E5DB E82500                     CALL    BEEP                    ; DO THE BEEP
E5DE E2FE              G2:      LOOP    G2                      ; DELAY BETWEEN BEEPS
E5E0 FECE                       DEC     DH                      ; ANY MORE TO DO
E5E2 75F5                       JNZ     G1                      ; DO IT
E5E4 803E120001                 CMP     MFG_TST,1               ; MFG TEST MODE?
E5E9 7506                       JNE     G3                      ; YES - CONTINUE BEEPING SPEAKER
E5EB B0CD                       MOV     AL,0CDH                 ; STOP BLINKING LED
E5ED E661                       OUT     PORT_B,AL
E5EF EBE8                       JMP     SHORT G1
E5F1                   G3:                                      ; SHORT_BEEP:
E5F1 B301                       MOV     BL,1                    ; COUNTER FOR A SHORT BEEP
E5F3 E80D00                     CALL    BEEP                    ; DO THE SOUND
E5F6                   G4:
E5F6 E2FE                       LOOP    G4                      ; DELAY BETWEEN BEEPS
E5F8 FECA                       DEC     DL                      ; DONE WITH SHORTS
E5FA 75F5                       JNZ     G3                      ; DO SOME MORE
E5FC                   G5:
E5FC E2FE                       LOOP    G5                      ; LONG DELAY BEFORE RETURN
E5FE                   G6:
E5FE E2FE                       LOOP    G6
E600 1F                         POP     DS                      ; RESTORE ORIG CONTENTS OF DS
E601 9D                         POPF                            ; RESTORE FLAGS TO ORIG SETTINGS
E602 C3                         RET                             ; RETURN TO CALLER
                       ERR_BEEP         ENDP

                       ;----- ROUTINE TO SOUND BEEPER

E603                   BEEP     PROC    NEAR
E603 B0B6                       MOV     AL,10110110B            ; SEL TIM 2,LSB,MSB,BINARY
E605 E643                       OUT     TIMER+3,AL              ; WRITE THE TIMER MODE REG
E607 B83305                     MOV     AX,533H                 ; DIVISOR FOR 1000 HZ
E60A E642                       OUT     TIMER+2,AL              ; WRITE TIMER 2 CNT - LSB
E60C 8AC4                       MOV     AL,AH
E60E E642                       OUT     TIMER+2,AL              ; WRITE TIMER 2 CNT - MSB
E610 E461                       IN      AL,PORT_B               ; GET CURRENT SETTING OF PORT
E612 8AE0                       MOV     AH,AL                   ; SAVE THAT SETTING
E614 0C03                       OR      AL,03                   ; TURN SPEAKER ON
E616 E661                       OUT     PORT_B,AL
E618 2BC9                       SUB     CX,CX                   ; SET CNT TO WAIT 500 MS
E61A                   G7:
E61A E2FE                       LOOP    G7                      ; DELAY BEFORE TURNING OFF
E61C FECB                       DEC     BL                      ; DELAY CNT EXPIRED?
E61E 75FA                       JNZ     G7                      ; NO - CONTINUE BEEPING SPK
E620 8AC4                       MOV     AL,AH                   ; RECOVER VALUE OF PORT
E622 E661                       OUT     PORT_B,AL
E624 C3                         RET                             ; RETURN TO CALLER
                       BEEP     ENDP

                       ;------------------------------------------------
                       ; CONVERT AND PRINT ASCII CODE                  :
                       ;        AL MUST CONTAIN NUMBER TO BE CONVERTED. :
                       ;        AX AND BX DESTROYED.                   :
                       ;------------------------------------------------
E625                   XPC_BYTE         PROC    NEAR
E625 50                         PUSH    AX                      ; RESAVE FOR LOW NIBBLE DISPLAY
E626 B104                       MOV     CL,4                    ; SHIFT COUNT
E628 D2E8                       SHR     AL,CL                   ; NIBBLE SWAP
E62A E80300                     CALL    XLAT_PR                 ; DO THE HIGH NIBBLE DISPLAY
E62D 58                         POP     AX                      ; RECOVER THE NIBBLE
E62E 240F                       AND     AL,0FH                  ; ISOLATE TO LOW NIBBLE
                                                                ; FALL INTO LOW NIBBLE CONVERSION
```

Appendix A

```
LOC OBJ                  SOURCE

E630                     XLAT_PR PROC     NEAR                ; CONVERT 00-0F TO ASCII CHARACTER
E630 0490                        ADD      AL,090H             ; ADD FIRST CONVERSION FACTOR
E632 27                          DAA                          ; ADJUST FOR NUMERIC AND ALPHA RANGE
E633 1440                        ADC      AL,040H             ; ADD CONVERSION AND ADJUST LOW NIBBLE
E635 27                          DAA                          ; ADJUST HI NIBBLE TO ASCII RANGE
E636                     PRT_HEX PROC     NEAR
E636 B40E                        MOV      AH,14               ; DISPLAY CHAR. IN AL
E638 B700                        MOV      BH,0
E63A CD10                        INT      10H                 ; CALL VIDEO_IO
E63C C3                          RET
                         PRT_HEX ENDP
                         XLAT_PR ENDP
                         XPC_BYTE         ENDP

E63D                     F4      LABEL    WORD                ; PRINTER SOURCE TABLE
E63D BC03                        DW       3BCH
E63F 7803                        DW       378H
E641 7802                        DW       278H
E643                     F4E     LABEL    WORD

                         ;-----------------------------------------------------------------------
                         ;       THIS PROCEDURE WILL SEND A SOFTWARE RESET TO THE KEYBOARD.     :
                         ;       SCAN CODE 'AA' SHOULD BE RETURNED TO THE CPU.                  :
                         ;-----------------------------------------------------------------------
E643                     KBD_RESET        PROC    NEAR
E643 B00C                        MOV      AL,0CH              ; SET KBD CLK LINE LOW
E645 E661                        OUT      PORT_B,AL           ; WRITE 8255 PORT B
E647 B95629                      MOV      CX,10582            ; HOLD KBD CLK LOW FOR 20 MS
E64A                     G8:
E64A E2FE                        LOOP     G8                  ; LOOP FOR 20 MS
E64C B0CC                        MOV      AL,0CCH             ; SET CLK, ENABLE LINES HIGH
E64E E661                        OUT      PORT_B,AL
E650                     SP_TEST:                             ; ENTRY FOR MANUFACTURING TEST 2
E650 B04C                        MOV      AL,4CH              ; SET KBD CLK HIGH, ENABLE LOW
E652 E661                        OUT      PORT_B,AL
E654 B0FD                        MOV      AL,0FDH             ; ENABLE KEYBOARD INTERRUPTS
E656 E621                        OUT      INTA01,AL           ; WRITE 8259 IMR
E658 FB                          STI                          ; ENABLE SYSTEM INTERRUPTS
E659 B400                        MOV      AH,0                ; RESET INTERRUPT INDICATOR
E65B 2BC9                        SUB      CX,CX               ; SETUP INTERRUPT TIMEOUT CNT
E65D                     G9:
E65D F6C4FF                      TEST     AH,0FFH             ; DID A KEYBOARD INTR OCCUR?
E660 7502                        JNZ      G10                 ; YES - READ SCAN CODE RETURNED
E662 E2F9                        LOOP     G9                  ; NO - LOOP TILL TIMEOUT
E664                     G10:
E664 E460                        IN       AL,PORT_A           ; READ KEYBOARD SCAN CODE
E666 8AD8                        MOV      BL,AL               ; SAVE SCAN CODE JUST READ
E668 B0CC                        MOV      AL,0CCH             ; CLEAR KEYBOARD
E66A E661                        OUT      PORT_B,AL
E66C C3                          RET                          ; RETURN TO CALLER
                         KBD_RESET        ENDP

                         ;-----------------------------------------------------------------------
                         ;       BLINK LED PROCEDURE FOR MFG BURN-IN AND RUN-IN TESTS           :
                         ;       IF LED IS ON, TURN IT OFF. IF OFF, TURN ON.                    :
                         ;-----------------------------------------------------------------------
E66D                     BLINK_INT        PROC    NEAR
E66D FB                          STI
E66E 50                          PUSH     AX                  ; SAVE AX REG CONTENTS
E66F E461                        IN       AL,PORT_B           ; READ CURRENT VAL OF PORT B
E671 8AE0                        MOV      AH,AL
E673 F6D0                        NOT      AL                  ; FLIP ALL BITS
E675 2440                        AND      AL,01000000B        ; ISOLATE CONTROL BIT
E677 80E4BF                      AND      AH,10111111B        ; MASK OUT OF ORIGINAL VAL
E67A 0AC4                        OR       AL,AH               ; OR NEW CONTROL BIT IN
E67C E661                        OUT      PORT_B,AL
E67E B020                        MOV      AL,EOI
E680 E620                        OUT      INTA00,AL
E682 58                          POP      AX                  ; RESTORE AX REG
E683 CF                          IRET
                         BLINK_INT        ENDP

                         ;----- CHECKSUM AND CALL INIT CODE IN OPTIONAL ROMS

E684                     ROM_CHECK        PROC    NEAR
E684 B840D0                      MOV      AX,DATA             ; SET ES=DATA
E687 8EC0                        MOV      ES,AX
```

```
LOC OBJ                  LINE    SOURCE

E689 2AE4                                SUB     AH,AH                   ; ZERO OUT AH
E68B 8A4702                              MOV     AL,[BX+2]               ; GET LENGTH INDICATOR
E68E B109                                MOV     CL,09H                  ; MULTIPLY BY 512
E690 D3E0                                SHL     AX,CL
E692 8BC8                                MOV     CX,AX                   ; SET COUNT
E694 51                                  PUSH    CX
E695 B104                                MOV     CL,4
E697 D3E8                                SHR     AX,CL
E699 03D0                                ADD     DX,AX                   ; SET POINTER TO NEXT MODULE
E69B 59                                  POP     CX

E69C E8B005                              CALL    ROS_CHECKSUM_CNT        ; DO CHECKSUM
E69F 7405                                JZ      ROM_CHECK_1
E6A1 E86501                              CALL    ROM_ERR                 ; PRINT ERROR INFO
E6A4 EB13                                JMP     SHORT ROM_CHECK_END
E6A6                             ROM_CHECK_1:
E6A6 52                                  PUSH    DX                      ; SAVE POINTER
E6A7 26C70600010300                      MOV     ES:IO_ROM_INIT,0003H    ; LOAD OFFSET
E6AE 268C1E0201                          MOV     ES:IO_ROM_SEG,DS        ; LOAD SEGMENT
E6B3 26FF1E0001                          CALL    DWORD PTR ES:IO_ROM_INIT        ; CALL INIT RTN.
E6B8 5A                                  POP     DX
E6B9                             ROM_CHECK_END:
E6B9 C3                                  RET
                                 ROM_CHECK       ENDP

                                 ;--------------------------------------------------------
                                 ; THIS SUBROUTINE WILL PRINT A MESSAGE ON THE DISPLAY   :
                                 ;                                                       :
                                 ; ENTRY REQUIREMENTS:                                   :
                                 ;       SI = OFFSET(ADDRESS) OF MESSAGE BUFFER          :
                                 ;       CX = MESSAGE BYTE COUNT                         :
                                 ;       MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS         :
                                 ;--------------------------------------------------------
E6BA                             P_MSG   PROC    NEAR
E6BA E88118                              CALL    DDS
E6BD 803E120001                          CMP     MFG_TST,1               ; MFG TEST MODE?
E6C2 7505                                JNE     G12                     ; NO - DISPLAY ERROR MSG
E6C4 B601                                MOV     DH,1                    ; YES - SETUP TO BEEP SPEAKER
E6C6 E906FF                              JMP     ERR_BEEP                ; YES - BEEP SPEAKER
E6C9                             G12:                                    ; WRITE_MSG:
E6C9 2E8A04                              MOV     AL,CS:[SI]              ; PUT CHAR IN AL
E6CC 46                                  INC     SI                      ; POINT TO NEXT CHAR
E6CD 50                                  PUSH    AX                      ; SAVE PRINT CHAR
E6CE E865FF                              CALL    PRT_HEX                 ; CALL VIDEO_IO
E6D1 58                                  POP     AX                      ; RECOVER PRINT CHAR
E6D2 3C0A                                CMP     AL,10                   ; WAS IT LINE FEED
E6D4 75F3                                JNE     G12                     ; NO,KEEP PRINTING STRING
E6D6 C3                                  RET
                                 P_MSG   ENDP

E6D7 20524F4D                    F3A     DB      ' ROM',13,10
E6DB 0D
E6DC 0A

E6DD                             D_EOI   PROC    NEAR
E6DD 50                                  PUSH    AX
E6DE B020                                MOV     AL,20H
E6E0 E620                                OUT     20H,AL
E6E2 58                                  POP     AX
E6E3 CF                                  IRET
                                 D_EOI   ENDP

                                 ;--- INT 19 ----------------------------------------------------
                                 ; BOOT STRAP LOADER                                             :
                                 ;       IF A 5 1/4" DISKETTE DRIVE IS AVAILABLE ON THE SYSTEM,  :
                                 ;       TRACK 0, SECTOR 1 IS READ INTO THE BOOT LOCATION        :
                                 ;       (SEGMENT 0, OFFSET 7C00) AND CONTROL IS TRANSFERRED     :
                                 ;       THERE.                                                  :
                                 ;                                                               :
                                 ;       IF THERE IS NO DISKETTE DRIVE, OR IF THERE IS A         :
                                 ;       HARDWARE ERROR CONTROL IS TRANSFERRED TO THE RESIDENT   :
                                 ;       BASIC ENTRY POINT.                                      :
                                 ;                                                               :
                                 ; IPL ASSUMPTIONS:                                              :
                                 ;       8255 PORT 60H BIT 0 = 1 IF IPL FROM DISKETTE            :
                                 ;---------------------------------------------------------------
                                         ASSUME  CS:CODE,DS:ABS0
```

Appendix A

```
LOC OBJ                  SOURCE

                         ;----- IPL WAS SUCCESSFUL

E6E4                     H4:
E6E4 EA007C0000                  JMP     BOOT_LOCN
E6F2                             ORG     0E6F2H
E6F2                     BOOT_STRAP      PROC    NEAR
E6F2 FB                          STI                             ; ENABLE INTERRUPTS
E6F3 2BC0                        SUB     AX,AX
E6F5 8ED8                        MOV     DS,AX

                         ;----- RESET DISKETTE PARAMETER TABLE VECTOR

E6F7 C7067800C7EF                MOV     WORD PTR DISK_POINTER,OFFSET DISK_BASE
E6FD 8C0E7A00                    MOV     WORD PTR DISK_POINTER+2,CS
E701 A11004                      MOV     AX,DATA_WORD[OFFSET EQUIP_FLAG] ; GET THE EQUIPMENT SWITCHES
E704 A801                        TEST    AL,1                    ; ISOLATE IPL SENSE SWITCH
E706 741E                        JZ      H3                      ; GO TO CASSETTE BASIC ENTRY POINT

                         ;----- MUST LOAD SYSTEM FROM DISKETTE -- CX HAS RETRY COUNT

E708 B90400                      MOV     CX,4                    ; SET RETRY COUNT
E70B                     H1:                                     ; IPL_SYSTEM
E70B 51                          PUSH    CX                      ; SAVE RETRY COUNT
E70C B400                        MOV     AH,0                    ; RESET THE DISKETTE SYSTEM
E70E CD13                        INT     13H                     ; DISKETTE_IO
E710 720F                        JC      H2                      ; IF ERROR, TRY AGAIN
E712 B80102                      MOV     AX,201H                 ; READ IN THE SINGLE SECTOR
E715 2BD2                        SUB     DX,DX
E717 8EC2                        MOV     ES,DX
E719 BB007C                      MOV     BX,OFFSET BOOT_LOCN
E71C B90100                      MOV     CX,1                    ; SECTOR 1, TRACK 0
E71F CD13                        INT     13H                     ; DISKETTE_IO
E721 59                  H2:     POP     CX                      ; RECOVER RETRY COUNT
E722 73C0                        JNC     H4                      ; CF SET BY UNSUCCESSFUL READ
E724 E2E5                        LOOP    H1                      ; DO IT FOR RETRY TIMES

                         ;----- UNABLE TO IPL FROM THE DISKETTE

E726                     H3:                                     ; CASSETTE_JUMP:
E726 CD18                        INT     18H                     ; USE INTERRUPT VECTOR TO GET TO BASIC
                         BOOT_STRAP      ENDP

                         ;-----INT 14-------------------------------------------------------------
                         ; RS232_IO                                                             :
                         ;       THIS ROUTINE PROVIDES BYTE STREAM I/O TO THE COMMUNICATIONS    :
                         ;       PORT ACCORDING TO THE PARAMETERS:                              :
                         ;       (AH)=0  INITIALIZE THE COMMUNICATIONS PORT                     :
                         ;               (AL) HAS PARAMETERS FOR INITIALIZATION                 :
                         ;                                                                      :
                         ;       7       6       5       4       3       2       1       0      :
                         ;       ----- BAUD RATE --      -PARITY--       STOPBIT --WORD LENGTH-- :
                         ;       000 - 110               X0 - NONE        0 - 1   10 - 7 BITS    :
                         ;       001 - 150               01 - ODD         1 - 2   11 - 8 BITS    :
                         ;       010 - 300               11 - EVEN                              :
                         ;       011 - 600                                                      :
                         ;       100 - 1200                                                     :
                         ;       101 - 2400                                                     :
                         ;       110 - 4800                                                     :
                         ;       111 - 9600                                                     :
                         ;                                                                      :
                         ;       ON RETURN, CONDITIONS SET AS IN CALL TO COMMO STATUS (AH=3)    :
                         ;       (AH)=1  SEND THE CHARACTER IN (AL) OVER THE COMMO LINE         :
                         ;               (AL) REGISTER IS PRESERVED                             :
                         ;               ON EXIT, BIT 7 OF AH IS SET IF THE ROUTINE WAS UNABLE  :
                         ;                       TO TRANSMIT THE BYTE OF DATA OVER THE LINE.    :
                         ;                       IF BIT 7 OF AH IS NOT SET, THE REMAINDER OF AH :
                         ;                       IS SET AS IN A STATUS REQUEST, REFLECTING THE  :
                         ;                       CURRENT STATUS OF THE LINE.                    :
                         ;       (AH)=2  RECEIVE A CHARACTER IN (AL) FROM COMMO LINE BEFORE     :
                         ;                       RETURNING TO CALLER                            :
                         ;               ON EXIT, AH HAS THE CURRENT LINE STATUS, AS SET BY THE :
                         ;                       THE STATUS ROUTINE, EXCEPT THAT THE ONLY BITS  :
                         ;                       LEFT ON ARE THE ERROR BITS (7,4,3,2,1)         :
                         ;                       IF AH HAS BIT 7 ON (TIME OUT) THE REMAINING    :
                         ;                       BITS ARE NOT PREDICTABLE.                      :
                         ;                       THUS, AH IS NON ZERO ONLY WHEN AN ERROR        :
```

```
LOC OBJ          LINE    SOURCE

                         ;                      OCCURRED.                                   :
                         ;          (AH)=3  RETURN THE CDMMO PORT STATUS IN (AX)            :
                         ;                  AH CONTAINS THE LINE STATUS                     :
                         ;                  BIT 7 = TIME OUT                                :
                         ;                  BIT 6 = TRANS SHIFT REGISTER EMPTY              :
                         ;                  BIT 5 = TRAH HDLDING REGISTER EMPTY             :
                         ;                  BIT 4 = BREAK DETECT                            :
                         ;                  BIT 3 = FRAMING ERROR                           :
                         ;                  BIT 2 = PARITY ERROR                            :
                         ;                  BIT 1 = DVERRUN ERROR                           :
                         ;                  BIT D = OATA READY                              :
                         ;                  AL CONTAINS THE MODEM STATUS                    :
                         ;                  BIT 7 = RECEIVED LINE SIGNAL DETECT             :
                         ;                  BIT 6 = RING INDICATDR                          :
                         ;                  BIT 5 = OATA SET READY                          :
                         ;                  BIT 4 = CLEAR TO SEND                           :
                         ;                  BIT 3 = DELTA RECEIVE LINE SIGNAL DETECT        :
                         ;                  BIT 2 = TRAILING EDGE RING DETECTOR             :
                         ;                  BIT 1 = DELTA DATA SET READY                    :
                         ;                  BIT 0 = DELTA CLEAR TD SEND                     :
                         ;                                                                  :
                         ;          (DX) = PARAMETER INDICATING WHICH RS232 CARD (0,1 ALLOWED)  :
                         ;                                                                  :
                         ; DATA AREA RS232_BASE CONTAINS THE BASE AODRESS OF THE 8250 ON THE  :
                         ;          CARO LDCATIDN 4D0H CONTAINS UP TO 4 RS232 ADDRESSES PDSSIBLE  :
                         ;          OATA AREA LABEL RS232_TIM_OUT (BYTE) CDNTAINS OUTER LOOP COUNT  :
                         ;          VALUE FOR TIMEOUT (DEFAULT=1)                           :
                         ; OUTPUT                                                           :
                         ;          AX MODIFIEO ACCOROING TO PARMS OF CALL                  :
                         ;          ALL OTHERS UNCHANGEO                                    :
                         ;--------------------------------------------------------------------
                                 ASSUME  CS:COOE,OS:OATA
E729                             DRG     0E729H
E729                     A1      LABEL   WORO                    ; TABLE OF INIT VALUE
E729 1704                        OW      1047                    ; 110 BAUD
E72B 0003                        OW      768                     ; 150
E72D 8001                        OW      384                     ; 300
E72F C000                        DW      192                     ; 600
E731 6000                        OW      96                      ; 1200
E733 3000                        OW      48                      ; 2400
E735 1800                        OW      24                      ; 4800
E737 0C00                        OW      12                      ; 9600

E739                     RS232_IO        PROC    FAR

                         ;----- VECTOR TO APPROPRIATE ROUTINE

E739 FB                          STI                             ; INTERRUPTS BACK DN
E73A 1E                          PUSH    OS                      ; SAVE SEGMENT
E73B 52                          PUSH    OX
E73C 56                          PUSH    SI
E73D 57                          PUSH    OI
E73E 51                          PUSH    CX
E73F 53                          PUSH    BX
E740 8BF2                        MDV     SI,DX                   ; RS232 VALUE TD SI
E742 8BFA                        MDV     DI,DX
E744 01E6                        SHL     SI,1                    ; WDRD DFFSET
E746 E8F517                      CALL    DDS
E749 8B14                        MOV     OX,RS232_BASE[SI]       ; GET BASE ADDRESS
E74B 0BD2                        OR      DX,OX                   ; TEST FOR 0 BASE ADDRESS
E74D 7413                        JZ      A3                      ; RETURN
E74F 0AE4                        OR      AH,AH                   ; TEST FOR (AH)=0
E751 7416                        JZ      A4                      ; COMMUN INIT
E753 FECC                        DEC     AH                      ; TEST FOR (AH)=1
E755 7445                        JZ      A5                      ; SEND AL
E757 FECC                        DEC     AN                      ; TEST FOR (AH)=2
E759 746A                        JZ      A12                     ; RECEIVE INTO AL
E75B                     A2:
E75B FECC                        DEC     AN                      ; TEST FDR (AH)=3
E75D 7503                        JNZ     A3
E75F E98300                      JMP     A18                     ; COMMUNICATION STATUS
E762                     A3:                                     ; RETURN FRDM RS232
E762 5B                          POP     BX
E763 59                          POP     CX
E764 5F                          POP     DI
E765 5E                          POP     SI
E766 5A                          POP     DX
```

Appendix A

```
LOC OBJ              SOURCE

E767 1F                      POP    DS
E768 CF                      IRET                          ; RETURN TO CALLER, NO ACTION

                     ;----- INITIALIZE THE COMMUNICATIONS PORT

E769                 A4:
E769 8AE0                    MOV    AH,AL                  ; SAVE INIT PARMS IN AH
E76B 83C203                  ADD    DX,3                   ; POINT TO 8250 CONTROL REGISTER
E76E B080                    MOV    AL,80H
E770 EE                      OUT    DX,AL                  ; SET DLAB=1

                     ;----- DETERMINE BAUD RATE DIVISOR

E771 8AD4                    MOV    DL,AH                  ; GET PARMS TO DL
E773 B104                    MOV    CL,4
E775 D2C2                    ROL    DL,CL
E777 81E20E00                AND    DX,0EH                 ; ISOLATE THEM
E77B BF29E7                  MOV    DI,OFFSET A1           ; BASE OF TABLE
E77E 03FA                    ADD    DI,DX                  ; PUT INTO INDEX REGISTER
E780 8B14                    MOV    DX,RS232_BASE[SI]      ; POINT TO HIGH ORDER OF DIVISOR
E782 42                      INC    DX
E783 2E8A4501                MOV    AL,CS:[DI]+1           ; GET HIGH ORDER OF DIVISOR
E787 EE                      OUT    DX,AL                  ; SET MS OF DIV TO 0
E788 4A                      DEC    DX
E789 2E8A05                  MOV    AL,CS:[DI]             ; GET LOW ORDER OF DIVISOR
E78C EE                      OUT    DX,AL                  ; SET LOW OF DIVISOR
E78D 83C203                  ADD    DX,3
E790 8AC4                    MOV    AL,AH                  ; GET PARMS BACK
E792 241F                    AND    AL,01FH                ; STRIP OFF THE BAUD BITS
E794 EE                      OUT    DX,AL                  ; LINE CONTROL TO 8 BITS
E795 4A                      DEC    DX
E796 4A                      DEC    DX
E797 B000                    MOV    AL,0
E799 EE                      OUT    DX,AL                  ; INTERRUPT ENABLES ALL OFF
E79A EB49                    JMP    SHORT A18              ; COM_STATUS

                     ;----- SEND CHARACTER IN (AL) OVER COMMO LINE

E79C                 A5:
E79C 50                      PUSH   AX                     ; SAVE CHAR TO SEND
E79D 83C204                  ADD    DX,4                   ; MODEM CONTROL REGISTER
E7A0 B003                    MOV    AL,3                   ; DTR AND RTS
E7A2 EE                      OUT    DX,AL                  ; DATA TERMINAL READY, REQUEST TO SEND
E7A3 42                      INC    DX                     ; MODEM STATUS REGISTER
E7A4 42                      INC    DX
E7A5 B730                    MOV    BH,30H                 ; DATA SET READY & CLEAR TO SEND
E7A7 E84800                  CALL   WAIT_FOR_STATUS        ; ARE BOTH TRUE
E7AA 740B                    JE     A9                     ; YES, READY TO TRANSMIT CHAR
E7AC                 A7:
E7AC 59                      POP    CX
E7AD 8AC1                    MOV    AL,CL                  ; RELOAD DATA BYTE
E7AF                 A8:
E7AF 80CC80                  OR     AH,80H                 ; INDICATE TIME OUT
E7B2 EBAE                    JMP    A3                     ; RETURN
E7B4                 A9:                                   ; CLEAR_TO_SEND
E7B4 4A                      DEC    DX                     ; LINE STATUS REGISTER
E7B5                 A10:                                  ; WAIT_SEND
E7B5 B720                    MOV    BH,20H                 ; IS TRANSMITTER READY
E7B7 E83800                  CALL   WAIT_FOR_STATUS        ; TEST FOR TRANSMITTER READY
E7BA 75F0                    JNZ    A7                     ; RETURN WITH TIME OUT SET
E7BC                 A11:                                  ; OUT_CHAR
E7BC 83EA05                  SUB    DX,5                   ; DATA PORT
E7BF 59                      POP    CX                     ; RECOVER IN CX TEMPORARILY
E7C0 8AC1                    MOV    AL,CL                  ; MOVE CHAR TO AL FOR OUT, STATUS IN AH
E7C2 EE                      OUT    DX,AL                  ; OUTPUT CHARACTER
E7C3 EB9D                    JMP    A3                     ; RETURN

                     ;----- RECEIVE CHARACTER FROM COMMO LINE

E7C5                 A12:
E7C5 83C204                  ADD    DX,4                   ; MODEM CONTROL REGISTER
E7C8 B001                    MOV    AL,1                   ; DATA TERMINAL READY
E7CA EE                      OUT    DX,AL
E7CB 42                      INC    DX                     ; MODEM STATUS REGISTER
E7CC 42                      INC    DX
E7CD                 A13:                                  ; WAIT_DSR
E7CD B720                    MOV    BH,20H                 ; DATA SET READY
```

```
LOC OBJ              LINE    SOURCE

E7CF E82000                          CALL    WAIT_FOR_STATUS        ; TEST FOR DSR
E7D2 750B                            JNZ     A8                     ; RETURN WITH ERROR
E7D4                         A15:                                   ; WAIT_DSR_END
E7D4 4A                              DEC     DX                     ; LINE STATUS REGISTER
E7D5                         A16:                                   ; WAIT_RECV
E7D5 B701                            MOV     BH,1                   ; RECEIVE BUFFER FULL
E7D7 E81800                          CALL    WAIT_FOR_STATUS        ; TEST FOR REC. BUFF. FULL
E7DA 75D3                            JNZ     A8                     ; SET TIME OUT ERROR
E7DC                         A17:                                   ; GET_CHAR
E7DC 80E41E                          AND     AH,00011110B           ; TEST FOR ERR CONDITIONS ON RECV CHAR
E7DF 8B14                            MOV     DX,RS232_BASE[SI]      ; DATA PORT
E7E1 EC                              IN      AL,DX                  ; GET CHARACTER FROM LINE
E7E2 E97DFF                          JMP     A3                     ; RETURN

                             ;----- COMMO PORT STATUS ROUTINE

E7E5                         A18:
E7E5 8B14                            MOV     DX,RS232_BASE[SI]
E7E7 83C205                          ADD     DX,5                   ; CONTROL PORT
E7EA EC                              IN      AL,DX                  ; GET LINE CONTROL STATUS
E7EB 8AE0                            MOV     AH,AL                  ; PUT IN AH FOR RETURN
E7ED 42                              INC     DX                     ; POINT TO MODEM STATUS REGISTER
E7EE EC                              IN      AL,DX                  ; GET MODEM CONTROL STATUS
E7EF E970FF                          JMP     A3                     ; RETURN
                             ;-----------------------------------------
                             ; WAIT FOR STATUS ROUTINE                :
                             ;                                        :
                             ; ENTRY:                                 :
                             ;         BH=STATUS BIT(S) TO LOOK FOR,  :
                             ;         DX=ADDR. OF STATUS REG         :
                             ; EXIT:                                  :
                             ;         ZERO FLAG ON  = STATUS FOUND   :
                             ;         ZERO FLAG OFF = TIMEOUT.       :
                             ;         AH=LAST STATUS READ            :
                             ;-----------------------------------------
E7F2                         WAIT_FOR_STATUS PROC    NEAR
E7F2 8A5D7C                          MOV     BL,RS232_TIM_OUT[DI]   ; LOAD OUTER LOOP COUNT
E7F5                         WFS0:
E7F5 2BC9                            SUB     CX,CX
E7F7                         WFS1:
E7F7 EC                              IN      AL,DX                  ; GET STATUS
E7F8 8AE0                            MOV     AH,AL                  ; MOVE TO AH
E7FA 22C7                            AND     AL,BH                  ; ISOLATE BITS TO TEST
E7FC 3AC7                            CMP     AL,BH                  ; EXACTLY = TO MASK
E7FE 7408                            JE      WFS_END                ; RETURN WITH ZERO FLAG ON
E800 E2F5                            LOOP    WFS1                   ; TRY AGAIN
E802 FECB                            DEC     BL
E804 75EF                            JNZ     WFS0
E806 0AFF                            OR      BH,BH                  ; SET ZERO FLAG OFF
E808                         WFS_END:
E808 C3                              RET
                             WAIT_FOR_STATUS ENDP
                             RS232_IO        ENDP

                             ;-------------------------------------------------------------------
                             ;        PRINT ADDRESS AND ERROR MESSAGE FOR ROM CHECKSUM ERRORS    :
                             ;-------------------------------------------------------------------
E809                         ROM_ERR PROC    NEAR
E809 52                              PUSH    DX                     ; SAVE POINTER
E80A 50                              PUSH    AX
E80B 8CDA                            MOV     DX,DS                  ; GET ADDRESS POINTER
E80D 81FA00C8                        CMP     DX,0C800H
E811 7E13                            JLE     ROM_ERR_BEEP           ; SPECIAL ERROR INDICATION
E813 8AC6                            MOV     AL,DH
E815 E80DFE                          CALL    XPC_BYTE               ; DISPLAY ADDRESS
E818 8AC2                            MOV     AL,DL
E81A E808FE                          CALL    XPC_BYTE
E81D BE07E6                          MOV     SI,OFFSET F3A          ; DISPLAY ERROR MSG
E820 E897FE                          CALL    P_MSG
E823                         ROM_ERR_END:
E823 58                              POP     AX
E824 5A                              POP     DX
E825 C3                              RET
E826                         ROM_ERR_BEEP:
E826 BA0201                          MOV     DX,0102H               ; BEEP 1 LONG, 2 SHORT
E829 E8A3FD                          CALL    ERR_BEEP
E82C EBF5                            JMP     SHORT ROM_ERR_END
```

Appendix A

```
LOC OBJ                 LINE    SOURCE

                                ROM_ERR ENDP

                                ;---- INT 16 -------------------------------------------------------
                                ; KEYBOARD I/O                                                     :
                                ;       THESE ROUTINES PROVIDE KEYBOARD SUPPORT                    :
                                ; INPUT                                                            :
                                ;       (AH)=0  READ THE NEXT ASCII CHARACTER STRUCK FROM THE KEYBOARD :
                                ;               RETURN THE RESULT IN (AL), SCAN CODE IN (AH)        :
                                ;       (AH)=1  SET THE Z FLAG TO INDICATE IF AN ASCII CHARACTER IS :
                                ;               AVAILABLE TO BE READ.                              :
                                ;               (ZF)=1 -- NO CODE AVAILABLE                        :
                                ;               (ZF)=0 -- CODE IS AVAILABLE                        :
                                ;               IF ZF = 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ :
                                ;               IS IN AX, AND THE ENTRY REMAINS IN THE BUFFER      :
                                ;       (AH)=2  RETURN THE CURRENT SHIFT STATUS IN AL REGISTER     :
                                ;               THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE :
                                ;               THE EQUATES FOR KB_FLAG                            :
                                ; OUTPUT                                                           :
                                ;       AS NOTED ABOVE, ONLY AX AND FLAGS CHANGED                  :
                                ;       ALL REGISTERS PRESERVED                                    :
                                ;-------------------------------------------------------------------
                                        ASSUME  CS:CODE,DS:DATA
E82E                                    ORG     0E82EH
E82E                            KEYBOARD_IO     PROC    FAR
E82E FB                                 STI                             ; INTERRUPTS BACK ON
E82F 1E                                 PUSH    DS                      ; SAVE CURRENT DS
E830 53                                 PUSH    BX                      ; SAVE BX TEMPORARILY
E831 E80A17                             CALL    DDS
E834 0AE4                               OR      AH,AH                   ; AH=0
E836 740A                               JZ      K1                      ; ASCII_READ
E838 FECC                               DEC     AH                      ; AH=1
E83A 741E                               JZ      K2                      ; ASCII_STATUS
E83C FECC                               DEC     AH                      ; AH=2
E83E 742B                               JZ      K3                      ; SHIFT_STATUS
E840 EB2C                               JMP     SHORT INT10_END         ; EXIT

                                ;----- READ THE KEY TO FIGURE OUT WHAT TO DO

E842                            K1:                                     ; ASCII READ
E842 FB                                 STI                             ; INTERRUPTS BACK ON DURING LOOP
E843 90                                 NOP                             ; ALLOW AN INTERRUPT TO OCCUR
E844 FA                                 CLI                             ; INTERRUPTS BACK OFF
E845 8B1E1A00                           MOV     BX,BUFFER_HEAD          ; GET POINTER TO HEAD OF BUFFER
E849 3B1E1C00                           CMP     BX,BUFFER_TAIL          ; TEST END OF BUFFER
E84D 74F3                               JZ      K1                      ; LOOP UNTIL SOMETHING IN BUFFER
E84F 8B07                               MOV     AX,[BX]                 ; GET SCAN CODE AND ASCII CODE
E851 E81D00                             CALL    K4                      ; MOVE POINTER TO NEXT POSITION
E854 891E1A00                           MOV     BUFFER_HEAD,BX          ; STORE VALUE IN VARIABLE
E858 EB14                               JMP     SHORT INT10_END         ; RETURN

                                ;----- ASCII STATUS

E85A                            K2:
E85A FA                                 CLI                             ; INTERRUPTS OFF
E85B 8B1E1A00                           MOV     BX,BUFFER_HEAD          ; GET HEAD POINTER
E85F 3B1E1C00                           CMP     BX,BUFFER_TAIL          ; IF EQUAL (Z=1) THEN NOTHING THERE
E863 8B07                               MOV     AX,[BX]
E865 FB                                 STI                             ; INTERRUPTS BACK ON
E866 5B                                 POP     BX                      ; RECOVER REGISTER
E867 1F                                 POP     DS                      ; RECOVER SEGMENT
E868 CA0200                             RET     2                       ; THROW AWAY FLAGS

                                ;----- SHIFT STATUS

E86B                            K3:
E86B A01700                             MOV     AL,KB_FLAG              ; GET THE SHIFT STATUS FLAGS
E86E                            INT10_END:
E86E 5B                                 POP     BX                      ; RECOVER REGISTER
E86F 1F                                 POP     DS                      ; RECOVER REGISTERS
E870 CF                                 IRET                            ; RETURN TO CALLER
                                KEYBOARD_IO     ENDP

                                ;----- INCREMENT A BUFFER POINTER

E871                            K4      PROC    NEAR
E871 43                                 INC     BX                      ; MOVE TO NEXT WORD IN LIST
E872 43                                 INC     BX
```

```
LOC OBJ            LINE    SOURCE

E873 3B1E8200                       CMP     BX,BUFFER_END           ; AY END OF BUFFER?
E877 7504                           JHE     K5                      ; NO, CONTINUE
E879 8B1E8000                       MOV     BX,BUFFER_STARY         ; YES, RESET TO BUFFER BEGINNING
E87D                        K5:
E87D C3                             REY
                            K4      ENDP

                            ;----- YABLE OF SHIFT KEYS AND MASK VALUES

E87E                        K6      LABEL   BYTE
E87E 52                             DB      INS_KEY                 ; INSERT KEY
E87F 3A                             DB      CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
E880 45
E881 46
E882 38
E883 1D
E884 2A                             DB      LEFY_KEY,RIGHT_KEY
E885 36
  0008                      K6L     EQU     $-K6

                            ;----- SHIFT MASK YABLE

E886                        K7      LABEL   BYTE
E886 80                             DB      IHS_SHIFT               ; INSERT MODE SHIFT
E887 40                             DB      CAPS_SHIFY,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
E888 20
E889 10
E88A 08
E88B 04
E88C 02                             DB      LEFT_SHIFT,RIGHT_SHIFT
E88D 01

                            ;----- SCAN CODE TABLES

E88E 1B                     K8              DB      27,-1,0,-1,-1,-1,30,-1
E88F FF
E890 00
E891 FF
E892 FF
E893 FF
E894 1E
E895 FF
E896 FF                                     DB      -1,-1,-1,31,-1,127,-1,17
E897 FF
E898 FF
E899 1F
E89A FF
E89B 7F
E89C FF
E89D 11
E89E 17                                     DB      23,5,18,20,25,21,9,15
E89F 05
E8A0 12
E8A1 14
E8A2 19
E8A3 15
E8A4 09
E8A5 0F
E8A6 10                                     DB      16,27,29,10,-1,1,19
E8A7 1B
E8A8 1D
E8A9 0A
E8AA FF
E8AB 01
E8AC 13
E8AD 04                                     DB      4,6,7,8,10,11,12,-1,-1
E8AE 06
E8AF 07
E8B0 08
E8B1 0A
E8B2 0B
E8B3 0C
E8B4 FF
E8B5 FF
E8B6 FF                                     DB      -1,-1,28,26,24,3,22,2
E8B7 FF
E8B8 1C
```

Appendix A

```
LOC OBJ              LINE     SOURCE

E8B9 1A
E8BA 1B
E8BB 03
E8BC 16
E8BD 02
E8BE 0E                                 DB      14,13,-1,-1,-1,-1,-1,-1
E8BF 0D
E8C0 FF
E8C1 FF
E8C2 FF
E8C3 FF
E8C4 FF
E8C5 FF
E8C6 20                                 DB      ' ',-1
E8C7 FF
                              ;----- CTL TABLE SCAN
E8C8                          K9      LABEL   BYTE
E8C8 5E                                 DB      94,95,96,97,98,99,100,101
E8C9 5F
E8CA 60
E8CB 61
E8CC 62
E8CD 63
E8CE 64
E8CF 65
E8D0 66                                 DB      102,103,-1,-1,119,-1,132,-1
E8D1 67
E8D2 FF
E8D3 FF
E8D4 77
E8D5 FF
E8D6 84
E8D7 FF
E8D8 73                                 DB      115,-1,116,-1,117,-1,118,-1
E8D9 FF
E8DA 74
E8DB FF
E8DC 75
E8DD FF
E8DE 76
E8DF FF
E8E0 FF                                 DB      -1
                              ;----- LC TABLE
E8E1                          K10     LABEL   BYTE
E8E1 1B                                 DB      01BH,'1234567890-=',08H,09H
E8E2 31323334353637
     3839302D3D
E8EE 08
E8EF 09
E8F0 71776572747975                     DB      'qwertyuiop[]',0DH,-1,'asdfghjkl;',027H
     696F705B5D
E8FC 0D
E8FD FF
E8FE 6173646667686A
     6B6C3B
E908 27
E909 60                                 DB      60H,-1,5CH,'zxcvbnm,./',-1,'*',-1,' '
E90A FF
E90B 5C
E90C 7A786376626E6D
     2C2E2F
E916 FF
E917 2A
E918 FF
E919 20
E91A FF                                 DB      -1
                              ;----- UC TABLE
E91B                          K11     LABEL   BYTE
E91B 1B                                 DB      27,'!@#$',37,05EH,'&*()_+',08H,0
E91C 21402324
E920 25
E921 5E
E922 262A28295F2B
E928 08
E929 00
E92A 51574552545955                     DB      'QWERTYUIOP{}',0DH,-1,'ASDFGHJKL:"'
     494F507B7D
```

```
LOC OBJ                 SOURCE

E936 0D
E937 FF
E938 4153444647484A
     4B4C3A22
E943 7E                                  DB      07EH,-1,'|ZXCVBNM<>?',-1,0,-1,' ',-1
E944 FF
E945 7C5A584356424E
     4D3C3E3F
E950 FF
E951 00
E952 FF
E953 20
E954 FF
                        ;----- UC TABLE SCAN
E955                    K12     LABEL   BYTE
E955 54                                 DB      84,85,86,87,88,89,90
E956 55
E957 56
E958 57
E959 58
E95A 59
E95B 5A
E95C 5B                                 DB      91,92,93
E95D 5C
E95E 5D
                        ;----- ALT TABLE SCAN
E95F                    K13     LABEL   BYTE
E95F 68                                 DB      104,105,106,107,108
E960 69
E961 6A
E962 6B
E963 6C
E964 6D                                 DB      109,110,111,112,113
E965 6E
E966 6F
E967 70
E968 71
                        ;----- NUM STATE TABLE
E969                    K14     LABEL   BYTE
E969 3738392D343536                     DB      '789-456+1230.'
     2B313233302E
                        ;----- BASE CASE TABLE
E976                    K15     LABEL   BYTE
E976 47                                 DB      71,72,73,-1,75,-1,77
E977 48
E978 49
E979 FF
E97A 4B
E97B FF
E97C 4D
E97D FF                                 DB      -1,79,80,81,82,83
E97E 4F
E97F 50
E980 51
E981 52
E982 53

                        ;----- KEYBOARD INTERRUPT ROUTINE

E987                            ORG     0E987H
E987                    KB_INT  PROC    FAR
E987 FB                         STI                     ; ALLOW FURTHER INTERRUPTS
E988 50                         PUSH    AX
E989 53                         PUSH    BX
E98A 51                         PUSH    CX
E98B 52                         PUSH    DX
E98C 56                         PUSH    SI
E98D 57                         PUSH    DI
E98E 1E                         PUSH    DS
E98F 06                         PUSH    ES
E990 FC                         CLD                     ; FORWARD DIRECTION
E991 E8AA15                     CALL    DDS
E994 E460                       IN      AL,KB_DATA      ; READ IN THE CHARACTER
E996 50                         PUSH    AX              ; SAVE IT
E997 E461                       IN      AL,KB_CTL       ; GET THE CONTROL PORT
E999 8AE0                       MOV     AH,AL           ; SAVE VALUE
E99B 0C80                       OR      AL,80H          ; RESET BIT FOR KEYBOARD
```

Appendix A

```
LOC OBJ                    SOURCE

E99D E661                          OUT     KB_CTL,AL
E99F 86E0                          XCHG    AH,AL                ; GET BACK ORIGIHAL CONTROL
E9A1 E661                          OUT     KB_CTL,AL            ; KB HAS BEEN RESET
E9A3 58                            POP     AX                   ; RECOVER SCAN COOE
E9A4 8AE0                          MOV     AN,AL                ; SAVE SCAH COOE IH AH ALSO

                           ;----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD

E9A6 3CFF                          CMP     AL,0FFH              ; IS THIS AH OVERRUN CHAR
E9A8 7503                          JNZ     K16                  ; NO, TEST FOR SHIFT KEY
E9AA E97A02                        JMP     K62                  ; BUFFER_FULL_BEEP

                           ;----- TEST FOR SHIFT KEYS

E9A0                       K16:                                 ; TEST_SHIFT
E9A0 247F                          AND     AL,07FH              ; TURH OFF THE BREAK BIT
E9AF 0E                            PUSN    CS
E9B0 07                            POP     ES                   ; ESTABLISN AOORESS OF SHIFT TABLE
E9B1 BF7EE8                        MOV     DI,OFFSET K6         ; SHIFT KEY TABLE
E9B4 B90800                        MOV     CX,K6L               ; LENGTH
E9B7 F2                            REPNE   SCASB                ; LOOK THROUGH THE TABLE FOR A MATCH
E9B8 AE
E9B9 8AC4                          MOV     AL,AH                ; RECOVER SCAN CODE
E9BB 7403                          JE      K17                  ; JUMP IF MATCH FOUND
E9BD E98500                        JMP     K25                  ; IF NO MATCH, THEN SHIFT NOT FOUND

                           ;----- SHIFT KEY FOUND

E9C0 81EF7FE8              K17:    SUB     DI,OFFSET K6+1       ; AOJUST PTR TO SCAH COOE MTCH
E9C4 2E8AA586E8                    MOV     AH,CS:K7[OI]         ; GET MASK INTO AH
E9C9 A880                          TEST    AL,80H               ; TEST FOR BREAK KEY
E9CB 7551                          JNZ     K23                  ; BREAK_SHIFT_FOUND

                           ;----- SHIFT MAKE FOUND, OETERMIHE SET OR TOGGLE

E9CD 80FC10                        CMP     AH,SCROLL_SHIFT
E9D0 7307                          JAE     K18                  ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY

                           ;----- PLAIH SHIFT KEY, SET SHIFT ON

E9D2 08261700                      OR      KB_FLAG,AH           ; TURH OH SHIFT BIT
E9D6 E98000                        JMP     K26                  ; INTERRUPT_RETURN

                           ;----- TOGGLEO SHIFT KEY, TEST FOR 1ST MAKE OR HOT

E9D9                       K18:                                 ; SHIFT-TOGGLE
E9D9 F606170004                    TEST    KB_FLAG, CTL_SHIFT   ; CHECK CTL SHIFT STATE
E9DE 7565                          JHZ     K25                  ; JUMP IF CTL STATE
E9E0 3C52                          CMP     AL, IHS_KEY          ; CHECK FOR INSERT KEY
E9E2 7522                          JHZ     K22                  ; JUMP IF NOT INSERT KEY
E9E4 F606170008                    TEST    KB_FLAG, ALT_SHIFT   ; CHECK FOR ALTERNATE SHIFT
E9E9 755A                          JNZ     K25                  ; JUMP IF ALTERHATE SHIFT
E9EB F606170020            K19:    TEST    KB_FLAG, NUM_STATE   ; CHECK FOR BASE STATE
E9F0 7500                          JNZ     K21                  ; JUMP IF NUM LOCK IS OH
E9F2 F606170003                    TEST    KB_FLAG, LEFT_SNIFT+ RIGHT_SHIFT
E9F7 7400                          JZ      K22                  ; JUMP IF BASE STATE

E9F9                       K20:                                 ; NUMERIC ZERO, NOT INSERT KEY
E9F9 B83052                        MOV     AX, 5230H            ; PUT OUT AH ASCII ZERO
E9FC E90601                        JMP     K57                  ; BUFFER_FILL
E9FF                       K21:                                 ; MIGHT BE NUMERIC
E9FF F606170003                    TEST    KB_FLAG, LEFT_SHIFT+ RIGHT_SHIFT
EA04 74F3                          JZ      K20                  ; JUMP NUMERIC, NOT INSERT

EA06                       K22:                                 ; SHIFT TOGGLE KEY HIT; PROCESS IT
EA06 84261800                      TEST    AH,KB_FLAG_1         ; IS KEY ALREAOY DEPRESSEO
EA0A 7540                          JNZ     K26                  ; JUMP IF KEY ALREADY OEPRESSEO
EA0C 08261800                      OR      KB_FLAG_1,AH         ; INDICATE THAT THE KEY IS OEPRESSED
EA10 30261700                      XOR     KB_FLAG,AH           ; TOGGLE THE SHIFT STATE
EA14 3C52                          CMP     AL,INS_KEY           ; TEST FOR 1ST MAKE OF IHSERT KEY
EA16 7541                          JNE     K26                  ; JUMP IF NOT INSERT KEY
EA18 B80052                        MOV     AX,INS_KEY*256       ; SET SCAH CODE IHTO AH, 0 INTO AL
EA1B E9B701                        JMP     K57                  ; PUT INTO OUTPUT BUFFER

                           ;----- BREAK SHIFT FOUND

EA1E                       K23:                                 ; BREAK-SHIFT-FOUND
```

```
LOC OBJ                 SOURCE

EA1E 80FC10                     CMP     AH,SCROLL_SHIFT         ; IS THIS A TOGGLE KEY
EA21 731A                       JAE     K24                     ; YES, HANDLE BREAK TOGGLE
EA23 F6D4                       NOT     AH                      ; INVERT MASK
EA25 20261700                   AND     KB_FLAG,AH              ; TURN OFF SHIFT BIT
EA29 3CB8                       CMP     AL,ALT_KEY+80H          ; IS THIS ALTERNATE SHIFT RELEASE
EA2B 752C                       JNE     K26                     ; INTERRUPT_RETURN

                ;----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER

EA2D A01900                     MOV     AL,ALT_INPUT
EA30 B400                       MOV     AH,0                    ; SCAN CODE OF 0
EA32 88261900                   MOV     ALT_INPUT,AH            ; ZERO OUT THE FIELD
EA36 3C00                       CMP     AL,0                    ; WAS THE INPUT=0
EA38 741F                       JE      K26                     ; INTERRUPT_RETURN
EA3A E9A101                     JMP     K58                     ; IT WASN'T, SO PUT IN BUFFER
EA3D                    K24:                                    ; BREAK-TOGGLE
EA3D F6D4                       NOT     AH                      ; INVERT MASK
EA3F 20261800                   AND     KB_FLAG_1,AH            ; INDICATE NO LONGER DEPRESSED
EA43 EB14                       JMP     SHORT K26               ; INTERRUPT_RETURN

                ;----- TEST FOR HOLD STATE

EA45                    K25:                                    ; NO-SHIFT-FOUND
EA45 3C80                       CMP     AL,80H                  ; TEST FOR BREAK KEY
EA47 7310                       JAE     K26                     ; NOTHING FOR BREAK CHARS FROM HERE ON
EA49 F606180008                 TEST    KB_FLAG_1,HOLD_STATE    ; ARE WE IN HOLD STATE
EA4E 7417                       JZ      K28                     ; BRANCH AROUND TEST IF NOT
EA50 3C45                       CMP     AL,NUM_KEY
EA52 7405                       JE      K26                     ; CAN'T END HOLD ON NUM_LOCK
EA54 80261800F7                 AND     KB_FLAG_1,NOT HOLD_STATE        ; TURN OFF THE HOLD STATE BIT
EA59                    K26:                                    ; INTERRUPT-RETURN
EA59 FA                         CLI                             ; TURN OFF INTERRUPTS
EA5A B020                       MOV     AL,EOI                  ; END OF INTERRUPT COMMAND
EA5C E620                       OUT     020H,AL                 ; SEND COMMAND TO INT CONTROL PORT
EA5E                    K27:                                    ; INTERRUPT-RETURN-NO-EOI
EA5E 07                         POP     ES
EA5F 1F                         POP     DS
EA60 5F                         POP     DI
EA61 5E                         POP     SI
EA62 5A                         POP     DX
EA63 59                         POP     CX
EA64 5B                         POP     BX
EA65 58                         POP     AX                      ; RESTORE STATE
EA66 CF                         IRET                            ; RETURN, INTERRUPTS BACK ON
                                                                ;  WITH FLAG CHANGE

                ;----- NOT IN  HOLD STATE, TEST FOR SPECIAL CHARS

EA67                    K28:                                    ; NO-HOLD-STATE
EA67 F606170008                 TEST    KB_FLAG,ALT_SHIFT       ; ARE WE IN ALTERNATE SHIFT
EA6C 7503                       JNZ     K29                     ; JUMP IF ALTERNATE SHIFT
EA6E E99100                     JMP     K38                     ; JUMP IF NOT ALTERNATE

                ;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)

EA71                    K29:                                    ; TEST-RESET
EA71 F606170004                 TEST    KB_FLAG,CTL_SHIFT       ; ARE WE IN CONTROL SHIFT ALSO
EA76 7433                       JZ      K31                     ; NO_RESET
EA78 3C53                       CMP     AL,DEL_KEY              ; SHIFT STATE IS THERE, TEST KEY
EA7A 752F                       JNE     K31                     ; NO_RESET

                ;----- CTL-ALT-DEL HAS BEEN FOUND, DO I/O CLEANUP

EA7C C70672003412               MOV     RESET_FLAG, 1234H       ; SET FLAG FOR RESET FUNCTION
EA82 EA5BE000F0                 JMP     RESET                   ; JUMP TO POWER ON DIAGNOSTICS

                ;----- ALT-INPUT-TABLE
EA87                    K30     LABEL   BYTE
EA87 52                         DB      82,79,80,81,75,76,77
EA88 4F
EA89 50
EA8A 51
EA8B 4B
EA8C 4C
EA8D 4D
EA8E 47                         DB      71,72,73                ; 10 NUMBERS ON KEYPAD
EA8F 48
```

```
LOC OBJ              SOURCE

EA90 49
                     ;----- SUPER-SHIFT-TABLE
EA91 10                      DB      16,17,18,19,20,21,22,23 ; A-Z TYPEWRITER CHARS
EA92 11
EA93 12
EA94 13
EA95 14
EA96 15
EA97 16
EA98 17
EA99 18                      DB      24,25,30,31,32,33,34,35
EA9A 19
EA9B 1E
EA9C 1F
EA9D 20
EA9E 21
EA9F 22
EAA0 23
EAA1 24                      DB      36,37,38,44,45,46,47,48
EAA2 25
EAA3 26
EAA4 2C
EAA5 2D
EAA6 2E
EAA7 2F
EAA8 30
EAA9 31                      DB      49,50
EAAA 32

                     ;----- IN ALTERNATE SHIFT, RESET NOT FOUND

EAAB                 K31:                                    ; NO-RESET
EAAB 3C39                    CMP     AL,57                   ; TEST FOR SPACE KEY
EAAD 7505                    JNE     K32                     ; NOT THERE
EAAF B020                    MOV     AL,' '                  ; SET SPACE CHAR
EAB1 E92101                  JMP     K57                     ; BUFFER_FILL

                     ;----- LOOK FOR KEY PAD ENTRY

EAB4                 K32:                                    ; ALT-KEY-PAD
EAB4 BF87EA                  MOV     DI,OFFSET K30           ; ALT-INPUT-TABLE
EAB7 B90A00                  MOV     CX,10                   ; LOOK FOR ENTRY USING KEYPAD
EABA F2                      REPNE   SCASB                   ; LOOK FOR MATCH
EABB AE
EABC 7512                    JNE     K33                     ; NO_ALT_KEYPAD
EABE 81EF88EA                SUB     DI,OFFSET K30+1         ; DI NOW HAS ENTRY VALUE
EAC2 A01900                  MOV     AL,ALT_INPUT            ; GET THE CURRENT BYTE
EAC5 B40A                    MOV     AH,10                   ; MULTIPLY BY 10
EAC7 F6E4                    MUL     AH
EAC9 03C7                    ADD     AX,DI                   ; ADD IN THE LATEST ENTRY
EACB A21900                  MOV     ALT_INPUT,AL            ; STORE IT AWAY
EACE EB89                    JMP     K26                     ; THROW AWAY THAT KEYSTROKE

                     ;----- LOOK FOR SUPERSHIFT ENTRY

EAD0                 K33:                                    ; NO-ALT-KEYPAD
EAD0 C606190000              MOV     ALT_INPUT,0             ; ZERO ANY PREVIOUS ENTRY INTO INPUT
EAD5 B91A00                  MOV     CX,26                   ; DI,ES ALREADY POINTING
EAD8 F2                      REPNE   SCASB                   ; LOOK FOR MATCH IN ALPHABET
EAD9 AE
EADA 7505                    JNE     K34                     ; NOT FOUND, FUNCTION KEY OR OTHER
EADC B000                    MOV     AL,0                    ; ASCII CODE OF ZERO
EADE E9F400                  JMP     K57                     ; PUT IT IN THE BUFFER

                     ;----- LOOK FOR TOP ROW OF ALTERNATE SHIFT

EAE1                 K34:                                    ; ALT-TOP-ROW
EAE1 3C02                    CMP     AL,2                    ; KEY WITH '1' ON IT
EAE3 720C                    JB      K35                     ; NOT ONE OF INTERESTING KEYS
EAE5 3C0E                    CMP     AL,14                   ; IS IT IN THE REGION
EAE7 7308                    JAE     K35                     ; ALT-FUNCTION
EAE9 80C476                  ADD     AH,118                  ; CONVERT PSUEDO SCAN CODE TO RANGE
EAEC B000                    MOV     AL,0                    ; INDICATE AS SUCH
EAEE E9E400                  JMP     K57                     ; BUFFER_FILL

                     ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES

```

```
LOC OBJ                SOURCE

EAF1                   K35:                                     ; ALT-FUNCTION
EAF1 3C3B                      CMP     AL,59                    ; TEST FOR IN TABLE
EAF3 7303                      JAE     K37                      ; ALT-CONTINUE
EAF5                   K36:                                     ; CLOSE-RETURN
EAF5 E961FF                    JMP     K26                      ; IGNORE THE KEY
EAF8                   K37:                                     ; ALT-CONTINUE
EAF8 3C47                      CMP     AL,71                    ; IN KEYPAD REGION
EAFA 73F9                      JAE     K36                      ; IF SO, IGNORE
EAFC BB5FE9                    MOV     BX,OFFSET K13            ; ALT SHIFT PSEUDO SCAN TABLE
EAFF E91B01                    JMP     K63                      ; TRANSLATE THAT

                       ;----- NOT IN ALTERNATE SHIFT

EB02                   K38:                                     ; NOT-ALT-SHIFT
EB02 F606170004                TEST    KB_FLAG,CTL_SHIFT        ; ARE WE IN CONTROL SHIFT
EB07 7458                      JZ      K44                      ; NOT-CTL-SHIFT

                       ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
                       ;----- TEST FOR BREAK AND PAUSE KEYS

EB09 3C46                      CMP     AL,SCROLL_KEY            ; TEST FOR BREAK
EB0B 7518                      JNE     K39                      ; NO-BREAK
EB0D 8B1E8000                  MOV     BX,BUFFER_START          ; RESET BUFFER TO EMPTY
EB11 891E1A00                  MOV     BUFFER_HEAD,BX
EB15 891E1C00                  MOV     BUFFER_TAIL,BX
EB19 C606710080                MOV     BIOS_BREAK,80H           ; TURN ON BIOS_BREAK BIT
EB1E CD1B                      INT     1BH                      ; BREAK INTERRUPT VECTOR
EB20 2BC0                      SUB     AX,AX                    ; PUT OUT DUMMY CHARACTER
EB22 E9B000                    JMP     K57                      ; BUFFER_FILL
EB25                   K39:                                     ; NO-BREAK
EB25 3C45                      CMP     AL,NUM_KEY               ; LOOK FOR PAUSE KEY
EB27 7521                      JNE     K41                      ; NO-PAUSE
EB29 800E180008                OR      KB_FLAG_1,HOLD_STATE     ; TURN ON THE HOLD FLAG
EB2E B020                      MOV     AL,EOI                   ; END OF INTERRUPT TO CONTROL PORT
EB30 E620                      OUT     020H,AL                  ; ALLOW FURTHER KEYSTROKE INTS

                       ;----- DURING PAUSE INTERVAL, TURN CRT BACK ON

EB32 803E490007                CMP     CRT_MODE,7               ; IS THIS BLACK AND WHITE CARD
EB37 7407                      JE      K40                      ; YES, NOTHING TO DO
EB39 BAD803                    MOV     DX,03D8H                 ; PORT FOR COLOR CARD
EB3C A06500                    MOV     AL,CRT_MODE_SET          ; GET THE VALUE OF THE CURRENT MODE
EB3F EE                        OUT     DX,AL                    ; SET THE CRT MODE, SO THAT CRT IS ON
EB40                   K40:                                     ; PAUSE-LOOP
EB40 F606180008                TEST    KB_FLAG_1,HOLD_STATE
EB45 75F9                      JNZ     K40                      ; LOOP UNTIL FLAG TURNED OFF
EB47 E914FF                    JMP     K27                      ; INTERRUPT_RETURN_NO_EOI
EB4A                   K41:                                     ; NO-PAUSE

                       ;----- TEST SPECIAL CASE KEY 55

EB4A 3C37                      CMP     AL,55
EB4C 7506                      JNE     K42                      ; NOT-KEY-55
EB4E B80072                    MOV     AX,114*256               ; START/STOP PRINTING SWITCH
EB51 E98100                    JMP     K57                      ; BUFFER_FILL

                       ;----- SET UP TO TRANSLATE CONTROL SHIFT

EB54                   K42:                                     ; NOT-KEY-55
EB54 BB8EE8                    MOV     BX,OFFSET K8             ; SET UP TO TRANSLATE CTL
EB57 3C3B                      CMP     AL,59                    ; IS IT IN TABLE
                                                                ; CTL-TABLE-TRANSLATE
EB59 7276                      JB      K56                      ; YES, GO TRANSLATE CHAR
EB5B                   K43:                                     ; CTL-TABLE-TRANSLATE
EB5B BBC8E8                    MOV     BX,OFFSET K9             ; CTL TABLE SCAN
EB5E E9BC00                    JMP     K63                      ; TRANSLATE_SCAN

                       ;----- NOT IN CONTROL SHIFT

EB61                   K44:                                     ; NOT-CTL-SHIFT
EB61 3C47                      CMP     AL,71                    ; TEST FOR KEYPAD REGION
EB63 732C                      JAE     K48                      ; HANDLE KEYPAD REGION
EB65 F606170003                TEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT
EB6A 745A                      JZ      K54                      ; TEST FOR SHIFT STATE

                       ;----- UPPER CASE, HANDLE SPECIAL CASES
```

```
LOC OBJ             SOURCE

EB6C 3C0F                   CMP     AL,15                   ; BACK TAB KEY
EB6E 7505                   JNE     K45                     ; NOT-BACK-TAB
EB70 B8000F                 MOV     AX,15*256               ; SET PSEUDO SCAN CODE
EB73 EB60                   JMP     SHORT K57               ; BUFFER_FILL
EB75                K45:                                    ; NOT-BACK-TAB
EB75 3C37                   CMP     AL,55                   ; PRINT SCREEN KEY
EB77 7509                   JNE     K46                     ; NOT-PRINT-SCREEN

                    ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION

EB79 B020                   MOV     AL,EOI                  ; END OF CURRENT INTERRUPT
EB7B E620                   OUT     020H,AL                 ;  SO FURTHER THINGS CAN HAPPEN
EB7D CD05                   INT     5H                      ; ISSUE PRINT SCREEN INTERRUPT
EB7F E90CFE                 JMP     K27                     ; GO BACK WITHOUT EOI OCCURRING
EB82                K46:                                    ; NOT-PRINT-SCREEN
EB82 3C3B                   CMP     AL,59                   ; FUNCTION KEYS
EB84 7206                   JB      K47                     ; NOT-UPPER-FUNCTION
EB86 BB55E9                 MOV     BX,OFFSET K12           ; UPPER CASE PSEUDO SCAN CODES
EB89 E99100                 JMP     K63                     ; TRANSLATE_SCAN
EB8C                K47:                                    ; NOT-UPPER-FUNCTION
EB8C BB1BE9                 MOV     BX,OFFSET K11           ; POINT TO UPPER CASE TABLE
EB8F EB40                   JMP     SHORT K56               ; OK, TRANSLATE THE CHAR

                    ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION

EB91                K48:                                    ; KEYPAD-REGION
EB91 F606170020             TEST    KB_FLAG,NUM_STATE       ; ARE WE IN NUM_LOCK
EB96 7520                   JNZ     K52                     ; TEST FOR SURE
EB98 F606170003             TEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT  ; ARE WE IN SHIFT STATE
EB9D 7520                   JNZ     K53                     ; IF SHIFTED, REALLY NUM STATE

                    ;----- BASE CASE FOR KEYPAD

EB9F                K49:                                    ; BASE-CASE
EB9F 3C4A                   CMP     AL,74                   ; SPECIAL CASE FOR A COUPLE OF KEYS
EBA1 740B                   JE      K50                     ; MINUS
EBA3 3C4E                   CMP     AL,78
EBA5 740C                   JE      K51
EBA7 2C47                   SUB     AL,71                   ; CONVERT ORIGIN
EBA9 BB76E9                 MOV     BX,OFFSET K15           ; BASE CASE TABLE
EBAC EB71                   JMP     SHORT K64               ; CONVERT TO PSEUDO SCAN
EBAE                K50:
EBAE B82D4A                 MOV     AX,74*256+'-'           ; MINUS
EBB1 EB22                   JMP     SHORT K57               ; BUFFER_FILL
EBB3                K51:
EBB3 B82B4E                 MOV     AX,78*256+'+'           ; PLUS
EBB6 EB10                   JMP     SHORT K57               ; BUFFER_FILL

                    ;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS

EBB8                K52:                                    ; ALMOST-NUM-STATE
EBB8 F606170003             TEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT
EBBD 75E0                   JNZ     K49                     ; SHIFTED TEMP OUT OF NUM STATE
EBBF                K53:                                    ; REALLY_NUM_STATE
EBBF 2C46                   SUB     AL,70                   ; CONVERT ORIGIN
EBC1 BB69E9                 MOV     BX,OFFSET K14           ; NUM STATE TABLE
EBC4 EB0B                   JMP     SHORT K56               ; TRANSLATE_CHAR

                    ;----- PLAIN OLD LOWER CASE

EBC6                K54:                                    ; NOT-SHIFT
EBC6 3C3B                   CMP     AL,59                   ; TEST FOR FUNCTION KEYS
EBC8 7204                   JB      K55                     ; NOT-LOWER-FUNCTION
EBCA B000                   MOV     AL,0                    ; SCAN CODE IN AH ALREADY
EBCC EB07                   JMP     SHORT K57               ; BUFFER_FILL
EBCE                K55:                                    ; NOT-LOWER-FUNCTION
EBCE BBE1E8                 MOV     BX,OFFSET K10           ; LC TABLE

                    ;----- TRANSLATE THE CHARACTER

EBD1                K56:                                    ; TRANSLATE-CHAR
EBD1 FEC8                   DEC     AL                      ; CONVERT ORIGIN
EBD3 2E07                   XLAT    CS:K11                  ; CONVERT THE SCAN CODE TO ASCII

                    ;----- PUT CHARACTER INTO BUFFER

EBD5                K57:                                    ; BUFFER-FILL
```

```
LOC OBJ          LINE    SOURCE

EBD5 3CFF                        CMP     AL,-1                   ; IS THIS AN IGNORE CHAR
EBD7 741F                        JE      K59                     ; YES, DO NOTHING WITH IT
EBD9 80FCFF                      CMP     AH,-1                   ; LOOK FOR -1 PSEUDO SCAN
EBDC 741A                        JE      K59                     ; NEAR_INTERRUPT_RETURN

                         ;----- HANDLE THE CAPS LOCK PROBLEM

EBDE                     K58:                                    ; BUFFER-FILL-NOTEST
EBDE F606170040                  TEST    KB_FLAG,CAPS_STATE      ; ARE WE IN CAPS LOCK STATE
EBE3 7420                        JZ      K61                     ; SKIP IF NOT

                         ;----- IN CAPS LOCK STATE

EBE5 F606170003                  TEST    KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT  ; TEST FOR SHIFT STATE
EBEA 740F                        JZ      K60                     ; IF NOT SHIFT, CONVERT LOWER TO UPPER

                         ;----- CONVERT ANY UPPER CASE TO LOWER CASE

EBEC 3C41                        CMP     AL,'A'                  ; FIND OUT IF ALPHABETIC
EBEE 7215                        JB      K61                     ; NOT_CAPS_STATE
EBF0 3C5A                        CMP     AL,'Z'
EBF2 7711                        JA      K61                     ; NOT_CAPS_STATE
EBF4 0420                        ADD     AL,'a'-'A'              ; CONVERT TO LOWER CASE
EBF6 EB0D                        JMP     SHORT K61               ; NOT_CAPS_STATE
EBF8                     K59:                                    ; NEAR-INTERRUPT-RETURN
EBF8 E95EFE                      JMP     K26                     ; INTERRUPT_RETURN

                         ;----- CONVERT ANY LOWER CASE TO UPPER CASE

EBFB                     K60:                                    ; LOWER-TO-UPPER
EBFB 3C61                        CMP     AL,'a'                  ; FIND OUT IF ALPHABETIC
EBFD 7206                        JB      K61                     ; NOT_CAPS_STATE
EBFF 3C7A                        CMP     AL,'z'
EC01 7702                        JA      K61                     ; NOT_CAPS_STATE
EC03 2C20                        SUB     AL,'a'-'A'              ; CONVERT TO UPPER CASE
EC05                     K61:                                    ; NOT-CAPS-STATE
EC05 8B1E1C00                    MOV     BX,BUFFER_TAIL          ; GET THE END POINTER TO THE BUFFER
EC09 8BF3                        MOV     SI,BX                   ; SAVE THE VALUE
EC0B E863FC                      CALL    K4                      ; ADVANCE THE TAIL
EC0E 3B1E1A00                    CMP     BX,BUFFER_HEAD          ; HAS THE BUFFER WRAPPED AROUND
EC12 7413                        JE      K62                     ; BUFFER_FULL_BEEP
EC14 8904                        MOV     [SI],AX                 ; STORE THE VALUE
EC16 891E1C00                    MOV     BUFFER_TAIL,BX          ; MOVE THE POINTER UP
EC1A E93CFE                      JMP     K26                     ; INTERRUPT_RETURN

                         ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES

EC1D                     K63:                                    ; TRANSLATE-SCAN
EC1D 2C3B                        SUB     AL,59                   ; CONVERT ORIGIN TO FUNCTION KEYS
EC1F                     K64:                                    ; TRANSLATE-SCAN-ORGD
EC1F 2E07                        XLAT    CS:K9                   ; CTL TABLE SCAN
EC21 8AE0                        MOV     AH,AL                   ; PUT VALUE INTO AH
EC23 B000                        MOV     AL,0                    ; ZERO ASCII CODE
EC25 EBAE                        JMP     K57                     ; PUT IT INTO THE BUFFER

                         KB_INT  ENDP

                         ;----- BUFFER IS FULL, SOUND THE BEEPER

EC27                     K62:                                    ; BUFFER-FULL-BEEP
EC27 B020                        MOV     AL,EOI                  ; END OF INTERRUPT COMMAND
EC29 E620                        OUT     20H,AL                  ; SEND COMMAND TO INT CONTROL PORT
EC2B BB8000                      MOV     BX,080H                 ; NUMBER OF CYCLES FOR 1/12 SECOND TONE
EC2E E461                        IN      AL,KB_CTL               ; GET CONTROL INFORMATION
EC30 50                          PUSH    AX                      ; SAVE
EC31                     K65:                                    ; BEEP-CYCLE
EC31 24FC                        AND     AL,0FCH                 ; TURN OFF TIMER GATE AND SPEAKER DATA
EC33 E661                        OUT     KB_CTL,AL               ; OUTPUT TO CONTROL
EC35 B94800                      MOV     CX,48H                  ; HALF CYCLE TIME FOR TONE
EC38                     K66:
EC38 E2FE                        LOOP    K66                     ; SPEAKER OFF
EC3A 0C02                        OR      AL,2                    ; TURN ON SPEAKER BIT
EC3C E661                        OUT     KB_CTL,AL               ; OUTPUT TO CONTROL
EC3E B94800                      MOV     CX,48H                  ; SET UP COUNT
EC41                     K67:
EC41 E2FE                        LOOP    K67                     ; ANOTHER HALF CYCLE
EC43 4B                          DEC     BX                      ; TOTAL TIME COUNT
```

Appendix A

```
LOC OBJ          LINE    SOURCE

EC44 75EB                        JNZ     K65                     ; OO ANOTHER CYCLE
EC46 58                          POP     AX                      ; RECOVER CONTROL
EC47 E661                        OUT     KB_CTL,AL               ; OUTPUT THE CONTROL
EC49 E912FE                      JMP     K27
                         ;----------------------------------------
                         ;         ROS CHECKSUM SUBROUTINE          :
                         ;----------------------------------------
EC4C                     ROS_CHECKSUM    PROC    NEAR            ; NEXT_ROS_MODULE
EC4C B90020                      MOV     CX,8192                 ; NUMBER OF BYTES TO ADD
EC4F                     ROS_CHECKSUM_CNT:                       ; ENTRY FOR OPTIONAL ROS TEST
EC4F 32C0                        XOR     AL,AL
EC51                     C26:
EC51 0207                        ADD     AL,DS:[BX]
EC53 43                          INC     BX                      ; POINT TO NEXT BYTE
EC54 E2FB                        LOOP    C26                     ; ADD ALL BYTES IN ROS MODULE
EC56 0AC0                        OR      AL,AL                   ; SUM = 0?
EC58 C3                          RET
                         ROS_CHECKSUM    ENDP

                         ;-- INT 13 --------------------------------------------------------------
                         ; OISKETTE I/O                                                          :
                         ;       THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4" OISKETTE ORIVES    :
                         ; INPUT                                                                 :
                         ;       (AH)=0  RESET OISKETTE SYSTEM                                   :
                         ;               HARD RESET TO NEC, PREPARE COMMAND, RECAL REQUIRED      :
                         ;               ON ALL ORIVES                                           :
                         ;       (AH)=1  READ THE STATUS OF THE SYSTEM INTO (AL)                 :
                         ;               OISKETTE_STATUS FROM LAST OPERATION IS USED             :
                         ;                                                                       :
                         ; REGISTERS FOR READ/WRITE/VERIFY/FORMAT                                :
                         ;       (DL) - ORIVE NUMBER (0-3 ALLOWEO, VALUE CHECKEO)                :
                         ;       (DH) - HEAD NUMBER (0-1 ALLOWEO, NOT VALUE CHECKED)             :
                         ;       (CH) - TRACK NUMBER (0-39, NOT VALUE CHECKEO)                   :
                         ;       (CL) - SECTOR NUMBER (1-8, NOT VALUE CHECKED,                   :
                         ;                               NOT USEO FOR FORMAT)                    :
                         ;       (AL) - NUMBER OF SECTORS ( MAX = 8, NOT VALUE CHECKEO, NOT USED :
                         ;                                       FOR FORMAT)                     :
                         ;       (ES:BX) - AOORESS OF BUFFER ( NOT REQUIREO FOR VERIFY)          :
                         ;                                                                       :
                         ;       (AH)=2  READ THE OESIREO SECTORS INTO MEMORY                    :
                         ;       (AH)=3  WRITE THE OESIREO SECTORS FROM MEMORY                   :
                         ;       (AH)=4  VERIFY THE OESIREO SECTORS                              :
                         ;       (AH)=5  FORMAT THE OESIREO TRACK                                :
                         ;               FOR THE FORMAT OPERATION, THE BUFFER POINTER (ES,BX)    :
                         ;               MUST POINT TO THE COLLECTION OF OESIREO AOORESS FIELDS  :
                         ;               FOR THE TRACK.  EACH FIELO IS COMPOSEO OF 4 BYTES,      :
                         ;               (C,H,R,N), WHERE C = TRACK NUMBER, H=HEAO NUMBER,       :
                         ;               R = SECTOR NUMBER, N= NUMBER OF BYTES PER SECTOR        :
                         ;               (00=128, 01=256, 02=512, 03=1024).  THERE MUST BE ONE   :
                         ;               ENTRY FOR EVERY SECTOR ON THE TRACK.  THIS INFORMATION  :
                         ;               IS USEO TO FINO THE REQUESTEO SECTOR OURING READ/WRITE  :
                         ;               ACCESS.                                                 :
                         ;                                                                       :
                         ; OATA VARIABLE -- OISK_POINTER                                         :
                         ;       OOUBLE WORO POINTER TO THE CURRENT SET OF OISKETTE PARAMETERS   :
                         ; OUTPUT                                                                :
                         ;       AH = STATUS OF OPERATION                                        :
                         ;               STATUS BITS ARE OEFINEO IN THE EQUATES FOR              :
                         ;               OISKETTE_STATUS VARIABLE IN THE OATA SEGMENT OF THIS    :
                         ;               MODULE.                                                 :
                         ;       CY = 0  SUCCESSFUL OPERATION (AH=0 ON RETURN)                   :
                         ;       CY = 1  FAILEO OPERATION (AH HAS ERROR REASON)                  :
                         ;       FOR READ/WRITE/VERIFY                                           :
                         ;               DS,BX,DX,CH,CL PRESERVED                                :
                         ;               AL = NUMBER OF SECTORS ACTUALLY READ                    :
                         ;               ***** AL MAY NOT BE CORRECT IF TIME OUT ERROR OCCURS    :
                         ;       NOTE:   IF AN ERROR IS REPORTEO BY THE OISKETTE CODE, THE       :
                         ;               APPROPRIATE ACTION IS TO RESET THE OISKETTE, THEN RETRY :
                         ;               THE OPERATION. ON READ ACCESSES, NO MOTOR START OELAY   :
                         ;               IS TAKEN, SO THAT THREE RETRIES ARE REQUIREO ON READS   :
                         ;               TO ENSURE THAT THE PROBLEM IS NOT DUE TO MOTOR          :
                         ;               START-UP.                                               :
                         ;-----------------------------------------------------------------------
                                 ASSUME  CS:CODE,DS:DATA,ES:DATA
EC59                             ORG     0EC59H
EC59                     OISKETTE_IO     PROC    FAR
EC59 FB                          STI                             ; INTERRUPTS BACK ON
```

```
LOC OBJ             LINE    SOURCE

EC5A 53                          PUSH   BX                      ; SAVE ADDRESS
EC5B 51                          PUSH   CX
EC5C 1E                          PUSH   DS                      ; SAVE SEGMENT REGISTER VALUE
EC5D 56                          PUSH   SI                      ; SAVE ALL REGISTERS DURING OPERATION
EC5E 57                          PUSH   DI
EC5F 55                          PUSH   BP
EC60 52                          PUSH   DX
EC61 8BEC                        MOV    BP,SP                   ; SET UP POINTER TO HEAD PARM
EC63 E8D812                      CALL   DDS
EC66 E81C00                      CALL   J1                      ; CALL THE REST TO ENSURE DS RESTORED
EC69 BB0400                      MOV    BX,4                    ; GET THE MOTOR WAIT PARAMETER
EC6C E8FD01                      CALL   GET_PARM
EC6F 88264000                    MOV    MOTOR_COUNT,AH          ; SET THE TIMER COUNT FOR THE MOTOR
EC73 8A264100                    MOV    AH,DISKETTE_STATUS      ; GET STATUS OF OPERATION
EC77 80FC01                      CMP    AH,1                    ; SET THE CARRY FLAG TO INDICATE
EC7A F5                          CMC                            ;  SUCCESS OR FAILURE
EC7B 5A                          POP    DX                      ; RESTORE ALL REGISTERS
EC7C 5D                          POP    BP
EC7D 5F                          POP    DI
EC7E 5E                          POP    SI
EC7F 1F                          POP    DS
EC80 59                          POP    CX
EC81 5B                          POP    BX                      ; RECOVER ADDRESS
EC82 CA0200                      RET    2                       ; THROW AWAY SAVED FLAGS
                        DISKETTE_IO     ENDP

EC85                    J1       PROC   NEAR
EC85 8AF0                        MOV    DH,AL                   ; SAVE # SECTORS IN DH
EC87 80263F007F                  AND    MOTOR_STATUS,07FH       ; INDICATE A READ OPERATION
EC8C 0AE4                        OR     AH,AH                   ; AH=0
EC8E 7427                        JZ     DISK_RESET
EC90 FECC                        DEC    AH                      ; AH=1
EC92 7473                        JZ     DISK_STATUS
EC94 C606410000                  MOV    DISKETTE_STATUS,0       ; RESET THE STATUS INDICATOR
EC99 80FA04                      CMP    DL,4                    ; TEST FOR DRIVE IN 0-3 RANGE
EC9C 7313                        JAE    J3                      ; ERROR IF ABOVE
EC9E FECC                        DEC    AH                      ; AH=2
ECA0 7469                        JZ     DISK_READ
ECA2 FECC                        DEC    AH                      ; AH=3
ECA4 7503                        JNZ    J2                      ; TEST_DISK_VERF
ECA6 E99500                      JMP    DISK_WRITE
ECA9                    J2:                                     ; TEST_DISK_VERF
ECA9 FECC                        DEC    AH                      ; AH=4
ECAB 7467                        JZ     DISK_VERF
ECAD FECC                        DEC    AH                      ; AH=5
ECAF 7467                        JZ     DISK_FORMAT
ECB1                    J3:                                     ; BAD_COMMAND
ECB1 C606410001                  MOV    DISKETTE_STATUS,BAD_CMD ; ERROR CODE, NO SECTORS TRANSFERRED
ECB6 C3                          RET                            ; UNDEFINED OPERATION
                        J1       ENDP

                        ;----- RESET THE DISKETTE SYSTEM

ECB7                    DISK_RESET      PROC    NEAR
ECB7 BAF203                      MOV    DX,03F2H                ; ADAPTER CONTROL PORT
ECBA FA                          CLI                            ; NO INTERRUPTS
ECBB A03F00                      MOV    AL,MOTOR_STATUS         ; WHICH MOTOR IS ON
ECBE B104                        MOV    CL,4                    ; SHIFT COUNT
ECC0 D2E0                        SAL    AL,CL                   ; MOVE MOTOR VALUE TO HIGH NYBBLE
ECC2 A820                        TEST   AL, 20H                 ; SELECT CORRESPONDING DRIVE
ECC4 750C                        JNZ    J5                      ; JUMP IF MOTOR ONE IS ON
ECC6 A840                        TEST   AL, 40H
ECC8 7506                        JNZ    J4                      ; JUMP IF MOTOR TWO IS ON
ECCA A880                        TEST   AL, 80H
ECCC 7406                        JZ     J6                      ; JUMP IF MOTOR ZERO IS ON
ECCE FEC0                        INC    AL
ECD0                    J4:
ECD0 FEC0                        INC    AL
ECD2                    J5:
ECD2 FEC0                        INC    AL
ECD4                    J6:
ECD4 0C08                        OR     AL,8                    ; TURN ON INTERRUPT ENABLE
ECD6 EE                          OUT    DX,AL                   ; RESET THE ADAPTER
ECD7 C6063E0000                  MOV    SEEK_STATUS,0           ; SET RECAL REQUIRED ON ALL DRIVES
ECDC C606410000                  MOV    DISKETTE_STATUS,0       ; SET OK STATUS FOR DISKETTE
ECE1 0C04                        OR     AL,4                    ; TURN OFF RESET
ECE3 EE                          OUT    DX,AL                   ; TURN OFF THE RESET
```

Appendix A

```
LOC OBJ                  SOURCE

ECE4 FB                          STI                             ; REENABLE THE INTERRUPTS
ECE5 E82A02                      CALL    CHK_STAT_2              ; DO SENSE INTERRUPT STATUS
                                                                 ;  FOLLOWING RESET
ECE8 A04200                      MOV     AL,NEC_STATUS           ; IGNORE ERROR RETURN AND DO OWN TEST
ECEB 3CC0                        CMP     AL,0C0H                 ; TEST FOR DRIVE READY TRANSITION
ECED 7406                        JZ      J7                      ; EVERYTHING OK
ECEF 800E410020                  OR      DISKETTE_STATUS,BAD_NEC ; SET ERROR CODE
ECF4 C3                          RET

                         ;----- SEND SPECIFY COMMAND TO NEC

ECF5                     J7:                                     ; DRIVE_READY
ECF5 B403                        MOV     AH,03H                  ; SPECIFY COMMAND
ECF7 E84701                      CALL    NEC_OUTPUT              ; OUTPUT THE COMMAND
ECFA BB0100                      MOV     BX,1                    ; FIRST BYTE PARM IN BLOCK
ECFD E86C01                      CALL    GET_PARM                ;  TO THE NEC CONTROLLER
ED00 BB0300                      MOV     BX,3                    ; SECOND BYTE PARM IN BLOCK
ED03 E86601                      CALL    GET_PARM                ;  TO THE NEC CONTROLLER
ED06                     J8:                                     ; RESET_RET
ED06 C3                          RET                             ; RETURN TO CALLER
                         DISK_RESET      ENDP

                         ;----- DISKETTE STATUS ROUTINE

ED07                     DISK_STATUS     PROC    NEAR
ED07 A04100                      MOV     AL,DISKETTE_STATUS
ED0A C3                          RET
                         DISK_STATUS     ENDP

                         ;----- DISKETTE READ

ED0B                     DISK_READ       PROC    NEAR
ED0B B046                        MOV     AL,046H                 ; READ COMMAND FOR DMA
ED0D                     J9:                                     ; DISK_READ_CONT
ED0D E8BB01                      CALL    DMA_SETUP               ; SET UP THE DMA
ED10 B4E6                        MOV     AH,0E6H                 ; SET UP RD COMMAND FOR NEC CONTROLLER
ED12 EB36                        JMP     SHORT RW_OPN            ; GO DO THE OPERATION
                         DISK_READ       ENDP

                         ;----- DISKETTE VERIFY

ED14                     DISK_VERF       PROC    NEAR
ED14 B042                        MOV     AL,042H                 ; VERIFY COMMAND FOR DMA
ED16 EBF5                        JMP     J9                      ; DO AS IF DISK READ
                         DISK_VERF       ENDP

                         ;----- DISKETTE FORMAT

ED18                     DISK_FORMAT     PROC    NEAR
ED18 800E3F0080                  OR      MOTOR_STATUS,80H        ; INDICATE WRITE OPERATION
ED1D B04A                        MOV     AL,04AH                 ; WILL WRITE TO THE DISKETTE
ED1F E8A601                      CALL    DMA_SETUP               ; SET UP THE DMA
ED22 B44D                        MOV     AH,04DH                 ; ESTABLISH THE FORMAT COMMAND
ED24 EB24                        JMP     SHORT RW_OPN            ; DO THE OPERATION
ED26                     J10:                                    ; CONTINUATION OF RW_OPN FOR FMT
ED26 BB0700                      MOV     BX,7                    ; GET THE
ED29 E84001                      CALL    GET_PARM                ;  BYTES/SECTOR VALUE TO NEC
ED2C BB0900                      MOV     BX,9                    ; GET THE
ED2F E83A01                      CALL    GET_PARM                ;  SECTORS/TRACK VALUE TO NEC
ED32 BB0F00                      MOV     BX,15                   ; GET THE
ED35 E83401                      CALL    GET_PARM                ;  GAP LENGTH VALUE TO NEC
ED38 BB1100                      MOV     BX,17                   ; GET THE FILLER BYTE
ED3B E9AB00                      JMP     J16                     ;  TO THE CONTROLLER
                         DISK_FORMAT     ENDP

                         ;----- DISKETTE WRITE ROUTINE

ED3E                     DISK_WRITE      PROC    NEAR
ED3E 800E3F0080                  OR      MOTOR_STATUS,80H        ; INDICATE WRITE OPERATION
ED43 B04A                        MOV     AL,04AH                 ; DMA WRITE COMMAND
ED45 E88001                      CALL    DMA_SETUP
ED48 B4C5                        MOV     AH,0C5H                 ; NEC COMMAND TO WRITE TO DISKETTE
                         DISK_WRITE      ENDP

                         ;----- ALLOW WRITE ROUTINE TO FALL INTO RW_OPN

                         ;-----------------------------------------------------------------
```

```
LOC OBJ                 SOURCE

                        ; RW_OPN                                                       :
                        ;         THIS ROUTINE PERFORMS THE READ/WRITE/VERIFY OPERATION   :
                        ;--------------------------------------------------------------
ED4A                    RW_OPN  PROC    NEAR
ED4A 7308                       JNC     J11                     ; TEST FOR DMA ERROR
ED4C C606410009                 MOV     DISKETTE_STATUS,DMA_BOUNDARY    ; SET ERROR
ED51 B000                       MOV     AL,0                    ; NO SECTORS TRANSFERRED
ED53 C3                         RET                             ; RETURN TO MAIN ROUTINE
ED54                    J11:                                    ; DO_RW_OPN
ED54 50                         PUSH    AX                      ; SAVE THE COMMAND

                        ;----- TURN ON THE MOTOR AND SELECT THE DRIVE

ED55 51                         PUSH    CX                      ; SAVE THE T/S PARMS
ED56 8ACA                       MOV     CL,DL                   ; GET DRIVE NUMBER AS SHIFT COUNT
ED58 B001                       MOV     AL,1                    ; MASK FOR DETERMINING MOTOR BIT
ED5A D2E0                       SAL     AL,CL                   ; SHIFT THE MASK BIT
ED5C FA                         CLI                             ; NO INTERRUPTS WHILE DETERMINING
                                                                ;  MOTOR STATUS
ED5D C6064000FF                 MOV     MOTOR_COUNT,0FFH        ; SET LARGE COUNT DURING OPERATION
ED62 84063F00                   TEST    AL,MOTOR_STATUS         ; TEST THAT MOTOR FOR OPERATING
ED66 7531                       JNZ     J14                     ; IF RUNNING, SKIP THE WAIT
ED68 80263F00F0                 AND     MOTOR_STATUS,0F0H       ; TURN OFF ALL MOTOR BITS
ED6D 08063F00                   OR      MOTOR_STATUS,AL         ; TURN ON THE CURRENT MOTOR
ED71 FB                         STI                             ; INTERRUPTS BACK ON
ED72 B010                       MOV     AL,10H                  ; MASK BIT
ED74 D2E0                       SAL     AL,CL                   ; DEVELOP BIT MASK FOR MOTOR ENABLE
ED76 0AC2                       OR      AL,DL                   ; GET DRIVE SELECT BITS IN
ED78 0C0C                       OR      AL,0CH                  ; NO RESET, ENABLE DMA/INT
ED7A 52                         PUSH    DX                      ; SAVE REG
ED7B BAF203                     MOV     DX,03F2H                ; CONTROL PORT ADDRESS
ED7E EE                         OUT     DX,AL
ED7F 5A                         POP     DX                      ; RECOVER REGISTERS

                        ;----- WAIT FOR MOTOR IF WRITE OPERATION

ED80 F6063F0080                 TEST    MOTOR_STATUS,80H        ; IS THIS A WRITE
ED85 7412                       JZ      J14                     ; NO, CONTINUE WITHOUT WAIT
ED87 BB1400                     MOV     BX,20                   ; GET THE MOTOR WAIT
ED8A E8DF00                     CALL    GET_PARM                ;  PARAMETER
ED8D 0AE4                       OR      AH,AH                   ; TEST FOR NO WAIT
ED8F                    J12:                                    ; TEST_WAIT_TIME
ED8F 7408                       JZ      J14                     ; EXIT WITH TIME EXPIRED
ED91 2BC9                       SUB     CX,CX                   ; SET UP 1/8 SECOND LOOP TIME
ED93                    J13:
ED93 E2FE                       LOOP    J13                     ; WAIT FOR THE REQUIRED TIME
ED95 FECC                       DEC     AH                      ; DECREMENT TIME VALUE
ED97 EBF6                       JMP     J12                     ; ARE WE DONE YET
ED99                    J14:                                    ; MOTOR_RUNNING
ED99 FB                         STI                             ; INTERRUPTS BACK ON FOR BYPASS WAIT
ED9A 59                         POP     CX

                        ;----- DO THE SEEK OPERATION

ED9B E8DF00                     CALL    SEEK                    ; MOVE TO CORRECT TRACK
ED9E 58                         POP     AX                      ; RECOVER COMMAND
ED9F 8AFC                       MOV     BH,AH                   ; SAVE COMMAND IN BH
EDA1 B600                       MOV     DH,0                    ; SET NO SECTORS READ IN CASE OF ERROR
EDA3 724B                       JC      J17                     ; IF ERROR, THEN EXIT AFTER MOTOR OFF
EDA5 BEF0E090                   MOV     SI,OFFSET J17           ; DUMMY RETURN ON STACK FOR NEC_OUTPUT
EDA9 56                         PUSH    SI                      ;  SO THAT IT WILL RETURN TO MOTOR OFF
                                                                ;  LOCATION

                        ;----- SEND OUT THE PARAMETERS TO THE CONTROLLER

EDAA E89400                     CALL    NEC_OUTPUT              ; OUTPUT THE OPERATION COMMAND
EDAD 8A6601                     MOV     AH,[BP+1]               ; GET THE CURRENT HEAD NUMBER
EDB0 D0E4                       SAL     AH,1                    ; MOVE IT TO BIT 2
EDB2 D0E4                       SAL     AH,1
EDB4 80E404                     AND     AH,4                    ; ISOLATE THAT BIT
EDB7 0AE2                       OR      AH,DL                   ; OR IN THE DRIVE NUMBER
EDB9 E88500                     CALL    NEC_OUTPUT

                        ;----- TEST FOR FORMAT COMMAND

EDBC 80FF4D                     CMP     BH,04DH                 ; IS THIS A FORMAT OPERATION
EDBF 7503                       JNE     J15                     ; NO, CONTINUE WITH R/W/V
```

Appendix A

```
LOC OBJ                   SOURCE

EDC1 E962FF                        JMP     J1D                       ; IF 5D, HANDLE SPECIAL
EDC4                      J15:
EDC4 8AE5                          MOV     AH,CH                     ; CYLINDER NUMBER
EDC6 E87800                        CALL    NEC_OUTPUT
EDC9 8A6601                        MOV     AH,[BP+1]                 ; HEAD NUMBER FROM STACK
EDCC E87200                        CALL    NEC_OUTPUT
EDCF 8AE1                          MOV     AH,CL                     ; SECTOR NUMBER
EDD1 E86D00                        CALL    NEC_OUTPUT
EDD4 BB0700                        MOV     BX,7                      ; BYTES/SECTOR PARM FROM BLOCK
EDD7 E89200                        CALL    GET_PARM                  ;  TO THE NEC
EDDA BB0900                        MOV     BX,9                      ; EOT PARM FROM BLOCK
EDDD E88C00                        CALL    GET_PARM                  ;  TO THE NEC
EDE0 BB0B00                        MOV     BX,11                     ; GAP LENGTH PARM FROM BLOCK
EDE3 E88600                        CALL    GET_PARM                  ;  TO THE NEC
EDE6 BB0D00                        MOV     BX,13                     ; DTL PARM FROM BLOCK
EDE9                      J16:                                       ; RW_OPN_FINISH
EDE9 E88000                        CALL    GET_PARM                  ;  TO THE NEC
EDEC 5E                            POP     SI                        ; CAN NOW DISCARD THAT DUMMY
                                                                     ;  RETURN ADDRESS

                          ;----- LET THE OPERATION HAPPEN

EDED E84301                        CALL    WAIT_INT                  ; WAIT FOR THE INTERRUPT
EDF0                      J17:                                       ; MOTOR_OFF
EDF0 7245                          JC      J21                       ; LOOK FOR ERROR
EDF2 E87401                        CALL    RESULTS                   ; GET THE NEC STATUS
EDF5 723F                          JC      J20                       ; LOOK FOR ERROR

                          ;----- CHECK THE RESULTS RETURNED BY THE CONTROLLER

EDF7 FC                            CLD                               ; SET THE CORRECT DIRECTION
EDF8 BE4200                        MOV     SI,OFFSET NEC_STATUS      ; POINT TO STATUS FIELD
EDFB AC                            LODS    NEC_STATUS                ; GET ST0
EDFC 24C0                          AND     AL,0C0H                   ; TEST FOR NORMAL TERMINATION
EDFE 743B                          JZ      J22                       ; OPN_OK
EE00 3C40                          CMP     AL,040H                   ; TEST FOR ABNORMAL TERMINATION
EE02 7529                          JNZ     J18                       ; NOT ABNORMAL, BAD NEC

                          ;----- ABNORMAL TERMINATION, FIND OUT WHY

EE04 AC                            LODS    NEC_STATUS                ; GET ST1
EE05 D0E0                          SAL     AL,1                      ; TEST FOR EOT FOUND
EE07 B404                          MOV     AH,RECORD_NOT_FND
EE09 7224                          JC      J19                       ; RW_FAIL
EE0B D0E0                          SAL     AL,1
EE0D D0E0                          SAL     AL,1                      ; TEST FOR CRC ERROR
EE0F B410                          MOV     AH,BAD_CRC
EE11 721C                          JC      J19                       ; RW_FAIL
EE13 D0E0                          SAL     AL,1                      ; TEST FOR DMA OVERRUN
EE15 B408                          MOV     AH,BAD_DMA
EE17 7216                          JC      J19                       ; RW_FAIL
EE19 D0E0                          SAL     AL,1
EE1B D0E0                          SAL     AL,1                      ; TEST FOR RECORD NOT FOUND
EE1D B404                          MOV     AH,RECORD_NOT_FND
EE1F 720E                          JC      J19                       ; RW_FAIL
EE21 D0E0                          SAL     AL,1
EE23 B403                          MOV     AH,WRITE_PROTECT          ; TEST FOR WRITE_PROTECT
EE25 7208                          JC      J19                       ; RW_FAIL
EE27 D0E0                          SAL     AL,1                      ; TEST MISSING ADDRESS MARK
EE29 B402                          MOV     AH,BAD_ADDR_MARK
EE2B 7202                          JC      J19                       ; RW_FAIL

                          ;----- NEC MUST HAVE FAILED

EE2D                      J18:                                       ; RW-NEC-FAIL
EE2D B420                          MOV     AH,BAD_NEC
EE2F                      J19:                                       ; RW-FAIL
EE2F 08264100                      OR      DISKETTE_STATUS,AH
EE33 E87801                        CALL    NUM_TRANS                 ; HOW MANY WERE REALLY TRANSFERRED
EE36                      J20:                                       ; RW_ERR
EE36 C3                            RET                               ; RETURN TO CALLER
EE37                      J21:                                       ; RW_ERR_RES
EE37 E82F01                        CALL    RESULTS                   ; FLUSH THE RESULTS BUFFER
EE3A C3                            RET

                          ;----- OPERATION WAS SUCCESSFUL
```

```
LOC OBJ         LINE    SOURCE

EE3B                    J22:                                    ; OPN_OK
EE3B E87001                     CALL    NUM_TRANS               ; HOW MANY GOT MOVED
EE3E 32E4                       XOR     AH,AH                   ; NO ERRORS
EE40 C3                         RET
                        RW_OPN  ENDP
                        ;--------------------------------------------------------------------
                        ; NEC_OUTPUT                                                        :
                        ;       THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING  :
                        ;       FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL   :
                        ;       TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE       :
                        ;       AMOUNT OF TIME, SETTING THE DISKEYTE STATUS ON COMPLETION.     :
                        ; INPUT                                                             :
                        ;       (AH)    BYTE TO BE OUTPUT                                   :
                        ; OUTPUT                                                            :
                        ;       CY = 0  SUCCESS                                             :
                        ;       CY = 1  FAILURE -- DISKETTE STATUS UPDATED                  :
                        ;               IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL :
                        ;               HIGHER THAN THE CALLER OF NEC_OUTPUT.               :
                        ;               THIS REMOVES THE REQUIREMENT OF TESTING AFTER EVERY :
                        ;               CALL OF NEC_OUTPUT.                                 :
                        ;       (AL) DESTROYED                                              :
                        ;--------------------------------------------------------------------
EE41                    NEC_OUTPUT      PROC    NEAR
EE41 52                         PUSH    DX                      ; SAVE REGISTERS
EE42 51                         PUSH    CX
EE43 BAF403                     MOV     DX,03F4H                ; STATUS PORT
EE46 33C9                       XOR     CX,CX                   ; COUNT FOR TIME OUT
EE48                    J23:
EE48 EC                         IN      AL,DX                   ; GET STATUS
EE49 A840                       TEST    AL,040H                 ; TEST DIRECTION BIT
EE4B 740C                       JZ      J25                     ; DIRECTION OK
EE4D E2F9                       LOOP    J23
EE4F                    J24:                                    ; TIME_ERROR
EE4F 800E410080                 OR      DISKETTE_STATUS,TIME_OUT
EE54 59                         POP     CX
EE55 5A                         POP     DX                      ; SET ERROR CODE AND RESTORE REGS
EE56 58                         POP     AX                      ; DISCARD THE RETURN ADDRESS
EE57 F9                         STC                             ; INDICATE ERROR TO CALLER
EE58 C3                         RET
EE59                    J25:
EE59 33C9                       XOR     CX,CX                   ; RESET THE COUNT
EE5B                    J26:
EE5B EC                         IN      AL,DX                   ; GET THE STATUS
EE5C A880                       TEST    AL,080H                 ; IS IT READY
EE5E 7504                       JNZ     J27                     ; YES, GO OUTPUT
EE60 E2F9                       LOOP    J26                     ; COUNT DOWN AND TRY AGAIN
EE62 EBEB                       JMP     J24                     ; ERROR CONDITION
EE64                    J27:                                    ; OUTPUT
EE64 8AC4                       MOV     AL,AH                   ; GET BYTE TO OUTPUT
EE66 B2F5                       MOV     DL,0F5H                 ; DATA PORT (3F5)
EE68 EE                         OUT     DX,AL                   ; OUTPUT THE BYTE
EE69 59                         POP     CX                      ; RECOVER REGISTERS
EE6A 5A                         POP     DX
EE6B C3                         RET                             ; CY = 0 FROM TEST INSTRUCTION
                        NEC_OUTPUT      ENDP
                        ;--------------------------------------------------------------------
                        ; GET_PARM                                                          :
                        ;       THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE  :
                        ;       BLOCK POINTED AT BY THE DATA VARIABLE DISK_POINTER. A BYTE FROM :
                        ;       THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING :
                        ;       THE PARM IN BX                                              :
                        ; ENTRY --                                                          :
                        ;   BX = INDEX OF BYTE TO BE FETCHED * 2                            :
                        ;       IF THE LOW BIT OF BX IS ON, THE BYTE IS IMMEDIATELY OUTPUT  :
                        ;       TO THE NEC CONTROLLER                                       :
                        ; EXIT --                                                           :
                        ;   AH = THAT BYTE FROM BLOCK                                       :
                        ;--------------------------------------------------------------------
EE6C                    GET_PARM        PROC    NEAR
EE6C 1E                         PUSH    DS                      ; SAVE SEGMENT
EE6D 2BC0                       SUB     AX,AX                   ; ZERO TO AX
EE6F 8ED8                       MOV     DS,AX
                                ASSUME  DS:ABS0
EE71 C5367800                   LDS     SI,DISK_POINTER         ; POINT TO BLOCK
EE75 D1EB                       SHR     BX,1                    ; DIVIDE BX BY 2, AND SET FLAG
                                                                ;  FOR EXIT
EE77 8A20                       MOV     AH,[SI+BX]              ; GET THE WORD
```

```
LOC OBJ                  LINE    SOURCE

EE79 1F                                 POP     DS                      ; RESTORE SEGMENT
                                        ASSUME  DS:DATA
EE7A 72C5                               JC      NEC_OUTPUT              ; IF FLAG SET, OUTPUT TO CONTROLLER
EE7C C3                                 RET                             ; RETURN TO CALLER
                                 GET_PARM       ENDP
                                 ;----------------------------------------------------------------------
                                 ; SEEK                                                                 :
                                 ;      THIS ROUTINE WILL MOVE THE HEAD ON THE NAMEO ORIVE TO THE       :
                                 ;      NAMED TRACK.  IF THE ORIVE HAS NOT BEEN ACCESSED SINCE THE      :
                                 ;      DRIVE RESET COMMAND WAS ISSUEO, THE ORIVE WILL BE RECALIBRATED. :
                                 ; INPUT                                                                :
                                 ;      (DL) = DRIVE TO SEEK ON                                         :
                                 ;      (CH) = TRACK TO SEEK TO                                         :
                                 ; OUTPUT                                                               :
                                 ;      CT = 0 SUCCESS                                                  :
                                 ;      CT = 1 FAILURE -- OISKETTE_STATUS SET ACCOROINGLY               :
                                 ;      (AX) DESTROYEO                                                  :
                                 ;----------------------------------------------------------------------
EE7D                             SEEK   PROC    NEAR
EE7D B001                               MOV     AL,1                    ; ESTABLISH MASK FOR RECAL TEST
EE7F 51                                 PUSH    CX                      ; SAVE INPUT VALUES
EE80 8ACA                               MOV     CL,DL                   ; GET DRIVE VALUE INTO CL
EE82 D2C0                               ROL     AL,CL                   ; SHIFT IT BY THE DRIVE VALUE
EE84 59                                 POP     CX                      ; RECOVER TRACK VALUE
EE85 84063E00                           TEST    AL,SEEK_STATUS          ; TEST FOR RECAL REQUIRED
EE89 7513                               JNZ     J28                     ; NO_RECAL
EE8B 08063E00                           OR      SEEK_STATUS,AL          ; TURN ON THE NO RECAL BIT IN FLAG
EE8F 8407                               MOV     AH,07H                  ; RECALIBRATE COMMAND
EE91 E8ADFF                             CALL    NEC_OUTPUT
EE94 8AE2                               MOV     AH,OL
EE96 E8A8FF                             CALL    HEC_OUTPUT              ; OUTPUT THE ORIVE NUMBER
EE99 E87600                             CALL    CHK_STAT_2              ; GET THE INTERUPT AND SENSE INT STATUS
EE9C 7229                               JC      J32                     ; SEEK_ERROR

                                 ;----- DRIVE IS IH STNCH WITH CONTROLLER, SEEK TO TRACK

EE9E                             J28:
EE9E 840F                               MOV     AH,0FH                  ; SEEK COMMAND TO HEC
EEA0 E89EFF                             CALL    NEC_OUTPUT
EEA3 8AE2                               MOV     AH,OL                   ; ORIVE NUMBER
EEA5 E899FF                             CALL    NEC_OUTPUT
EEA8 8AE5                               MOV     AH,CH                   ; TRACK NUMBER
EEAA E894FF                             CALL    HEC_OUTPUT
EEAD E86200                             CALL    CHK_STAT_2              ; GET ENDIHG INTERRUPT AND
                                                                        ;  SENSE STATUS

                                 ;----- WAIT FOR HEAO SETTLE

EEB0 9C                                 PUSHF                           ; SAVE STATUS FLAGS
EEB1 881200                             MOV     BX,18                   ; GET HEAO SETTLE PARAMETER
EEB4 E8B5FF                             CALL    GET_PARM
EEB7 51                                 PUSH    CX                      ; SAVE REGISTER
EEB8                             J29:                                   ; HEAO_SETTLE
EEB8 892602                             MOV     CX,550                  ; 1 MS LOOP
EEBB 0AE4                               OR      AH,AH                   ; TEST FOR TIME EXPIREO
EEBD 7406                               JZ      J31
EEBF                             J30:
EEBF E2FE                               LOOP    J30                     ; OELAY FOR 1 MS
EEC1 FECC                               DEC     AH                      ; DECREMENT THE COUNT
EEC3 EBF3                               JMP     J29                     ; DO IT SOME MORE
EEC5                             J31:
EEC5 59                                 POP     CX                      ; RECOVER STATE
EEC6 90                                 POPF
EEC7                             J32:                                   ; SEEK_ERROR
EEC7 C3                                 RET                             ; RETURN TO CALLER
                                 SEEK   ENDP
                                 ;----------------------------------------------------------------------
                                 ; DMA_SETUP                                                            :
                                 ;      THIS ROUTINE SETS UP THE OMA FOR REAO/WRITE/VERIFT OPERATIONS.  :
                                 ; INPUT                                                                :
                                 ;      (AL) = MODE BTTE FOR THE OMA                                    :
                                 ;      (ES:BX) - ADDRESS TO REAO/WRITE THE OATA                        :
                                 ; OUTPUT                                                               :
                                 ;      (AX) DESTROYEO                                                  :
                                 ;----------------------------------------------------------------------
EEC8                             DMA_SETUP      PROC    HEAR
EEC8 51                                 PUSH    CX                      ; SAVE THE REGISTER
```

```
LOC OBJ                 SOURCE

EEC9 FA                         CLI                             ; NO MORE INTERRUPTS
EECA E60C                       OUT     DMA+12,AL               ; SET THE FIRST/LAST F/F
EECC 50                         PUSH    AX
EECD 58                         POP     AX
EECE E60B                       OUT     DMA+11,AL               ; OUTPUT THE MODE BYTE
EED0 8CC0                       MOV     AX,ES                   ; GET THE ES VALUE
EED2 B104                       MOV     CL,4                    ; SHIFT COUNT
EED4 D3C0                       ROL     AX,CL                   ; ROTATE LEFT
EED6 8AE8                       MOV     CH,AL                   ; GET HIGHEST NYBLE OF ES TO CH
EED8 24F0                       AND     AL,0F0H                 ; ZERO THE LOW NYBBLE FROM SEGMENT
EEDA 03C3                       ADD     AX,BX                   ; TEST FOR CARRY FROM ADDITION
EEDC 7302                       JNC     J33
EEDE FEC5                       INC     CH                      ; CARRY MEANS HIGH 4 BITS MUST BE INC
EEE0                    J33:
EEE0 50                         PUSH    AX                      ; SAVE START ADDRESS
EEE1 E604                       OUT     DMA+4,AL                ; OUTPUT LOW ADDRESS
EEE3 8AC4                       MOV     AL,AH
EEE5 E604                       OUT     DMA+4,AL                ; OUTPUT HIGH ADDRESS
EEE7 8AC5                       MOV     AL,CH                   ; GET HIGH 4 BITS
EEE9 240F                       AND     AL,0FH
EEEB E681                       OUT     081H,AL                 ; OUTPUT THE HIGH 4 BITS TO
                                                                ;  THE PAGE REGISTER

                        ;----- DETERMINE COUNT

EEED 8AE6                       MOV     AH,DH                   ; NUMBER OF SECTORS
EEEF 2AC0                       SUB     AL,AL                   ;  TIMES 256 INTO AX
EEF1 D1E8                       SHR     AX,1                    ; SECTORS * 128 INTO AX
EEF3 50                         PUSH    AX
EEF4 BB0600                     MOV     BX,6                    ; GET THE BYTES/SECTOR PARM
EEF7 E872FF                     CALL    GET_PARM
EEFA 8ACC                       MOV     CL,AH                   ; USE AS SHIFT COUNT (0=128, 1=256 ETC)
EEFC 58                         POP     AX
EEFD D3E0                       SHL     AX,CL                   ; MULTIPLY BY CORRECT AMOUNT
EEFF 48                         DEC     AX                      ; -1 FOR DMA VALUE
EF00 50                         PUSH    AX                      ; SAVE COUNT VALUE
EF01 E605                       OUT     DMA+5,AL                ; LOW BYTE OF COUNT
EF03 8AC4                       MOV     AL,AH
EF05 E605                       OUT     DMA+5,AL                ; HIGH BYTE OF COUNT
EF07 FB                         STI                             ; INTERRUPTS BACK ON
EF08 59                         POP     CX                      ; RECOVER COUNT VALUE
EF09 58                         POP     AX                      ; RECOVER ADDRESS VALUE
EF0A 03C1                       ADD     AX,CX                   ; ADD, TEST FOR 64K OVERFLOW
EF0C 59                         POP     CX                      ; RECOVER REGISTER
EF0D B002                       MOV     AL,2                    ; MODE FOR 8237
EF0F E60A                       OUT     DMA+10,AL               ; INITIALIZE THE DISKETTE CHANNEL
EF11 C3                         RET                             ; RETURN TO CALLER,
                                                                ;  CFL SET BY ABOVE IF ERROR
                        DMA_SETUP       ENDP
                        ;----------------------------------------------------------------------
                        ; CHK_STAT_2                                                          :
                        ;       THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER A           :
                        ;       RECALIBRATE, SEEK, OR RESET TO THE ADAPTER.                   :
                        ;       THE INTERRUPT IS WAITED FOR, THE INTERRUPT STATUS SENSED,     :
                        ;       AND THE RESULT RETURNED TO THE CALLER.                        :
                        ; INPUT                                                               :
                        ;       NONE                                                          :
                        ; OUTPUT                                                              :
                        ;       CY = 0 SUCCESS                                                :
                        ;       CY = 1 FAILURE -- ERROR IS IN DISKETTE_STATUS                 :
                        ;       (AX) DESTROYED                                                :
                        ;----------------------------------------------------------------------
EF12                    CHK_STAT_2      PROC    NEAR
EF12 E81E00                     CALL    WAIT_INT                ; WAIT FOR THE INTERRUPT
EF15 7214                       JC      J34                     ; IF ERROR, RETURN IT
EF17 B408                       MOV     AH,08H                  ; SENSE INTERRUPT STATUS COMMAND
EF19 E825FF                     CALL    NEC_OUTPUT
EF1C E84A00                     CALL    RESULTS                 ; READ IN THE RESULTS
EF1F 720A                       JC      J34                     ; CHK2_RETURN
EF21 A04200                     MOV     AL,NEC_STATUS           ; GET THE FIRST STATUS BTTE
EF24 2460                       AND     AL,060H                 ; ISOLATE THE BITS
EF26 3C60                       CMP     AL,060H                 ; TEST FOR CORRECT VALUE
EF28 7402                       JZ      J35                     ; IF ERROR, GO MARK IT
EF2A F8                         CLC                             ; GOOD RETURN
EF2B                    J34:
EF2B C3                         RET                             ; RETURN TO CALLER
EF2C                    J35:                                    ; CHK2_ERROR
```

Appendix A

```
LOC OBJ                SOURCE

EF2C 800E410040              OR      DISKETTE_STATUS,BAD_SEEK
EF31 F9                      STC                             ; ERROR RETURN CODE
EF32 C3                      RET
                       CHK_STAT_2      ENDP
                       ;------------------------------------------------------------------------
                       ; WAIT INT                                                               :
                       ;       THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR. A TIME OUT         :
                       ;       ROUTINE TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE     :
                       ;       RETURNED IF THE DRIVE IS NOT READY.                              :
                       ; INPUT                                                                  :
                       ;       NONE                                                             :
                       ; OUTPUT                                                                 :
                       ;       CY = 0 SUCCESS                                                   :
                       ;       CY = 1 FAILURE -- DISKETTE_STATUS IS SET ACCORDINGLY             :
                       ;       (AX) DESTROYED                                                   :
                       ;------------------------------------------------------------------------
EF33                   WAIT_INT        PROC    NEAR
EF33 F8                      STI                             ; TURN ON INTERRUPTS, JUST IN CASE
EF34 53                      PUSH    BX
EF35 51                      PUSH    CX                      ; SAVE REGISTERS
EF36 8302                    MOV     BL,2                    ; CLEAR THE COUNTERS
EF38 33C9                    XOR     CX,CX                   ; FOR 2 SECOND WAIT
EF3A                   J36:
EF3A F6063E0080              TEST    SEEK_STATUS,INT_FLAG    ; TEST FOR INTERRUPT OCCURRING
EF3F 750C                    JNZ     J37
EF41 E2F7                    LOOP    J36                     ; COUNT DOWN WHILE WAITING
EF43 FECB                    DEC     BL                      ; SECOND LEVEL COUNTER
EF45 75F3                    JNZ     J36
EF47 800E410080              OR      DISKETTE_STATUS,TIME_OUT        ; NOTHING HAPPENED
EF4C F9                      STC                             ; ERROR RETURN
EF4D                   J37:
EF4D 9C                      PUSHF                           ; SAVE CURRENT CARRY
EF4E 80263E007F              AND     SEEK_STATUS,NOT INT_FLAG        ; TURN OFF INTERRUPT FLAG
EF53 9D                      POPF                            ; RECOVER CARRY
EF54 59                      POP     CX
EF55 5B                      POP     BX                      ; RECOVER REGISTERS
EF56 C3                      RET                             ; GOOD RETURN CODE COMES
                                                             ;  FROM TEST INST
                       WAIT_INT        ENDP
                       ;--------------------------------------------------------
                       ; DISK_INT                                               :
                       ;       THIS ROUTINE HANDLES THE DISKETTE INTERRUPT      :
                       ;INPUT                                                   :
                       ;       NONE                                             :
                       ; OUTPUT                                                 :
                       ;       THE INTERRUPT FLAG IS SET IS SEEK_STATUS         :
                       ;--------------------------------------------------------
EF57                         ORG     0EF57H
EF57                   DISK_INT        PROC    FAR
EF57 FB                      STI                             ; RE ENABLE INTERRUPTS
EF58 1E                      PUSH    DS
EF59 50                      PUSH    AX
EF5A E8E10F                  CALL    DDS
EF5D 800E3E0080              OR      SEEK_STATUS,INT_FLAG
EF62 B020                    MOV     AL,20H                  ; END OF INTERRUPT MARKER
EF64 E620                    OUT     20H,AL                  ; INTERRUPT CONTROL PORT
EF66 58                      POP     AX
EF67 1F                      POP     DS                      ; RECOVER SYSTEM
EF68 CF                      IRET                            ; RETURN FROM INTERRUPT
                       DISK_INT        ENDP
                       ;------------------------------------------------------------------------
                       ; RESULTS                                                                :
                       ;       THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER HAS      :
                       ;       TO SAY FOLLOWING AN INTERRUPT.                                   :
                       ; INPUT                                                                  :
                       ;       NONE                                                             :
                       ; OUTPUT                                                                 :
                       ;       CY = 0  SUCCESSFUL TRANSFER                                      :
                       ;       CY = 1  FAILURE -- TIME OUT IN WAITING FOR STATUS                :
                       ;       NEC_STATUS AREA HAS STATUS BYTE LOADED INTO IT                   :
                       ;       (AH) DESTROYED                                                   :
                       ;------------------------------------------------------------------------
EF69                   RESULTS PROC    NEAR
EF69 FC                      CLD
EF6A BF4200                  MOV     DI,OFFSET NEC_STATUS    ; POINTER TO DATA AREA
EF6D 51                      PUSH    CX                      ; SAVE COUNTER
EF6E 52                      PUSH    DX
```

```
LOC OBJ           LINE    SOURCE

EF6F 53                           PUSH    BX
EF70 B307                         MOV     BL,7                    ; MAX STATUS BYTES

                          ;----- WAIT FOR REQUEST FOR MASTER

EF72                      J38:                                    ; INPUT_LOOP
EF72 33C9                         XOR     CX,CX                   ; COUNTER
EF74 BAF403                       MOV     DX,03F4H                ; STATUS PORT
EF77                      J39:                                    ; WAIT FOR MASTER
EF77 EC                           IN      AL,DX                   ; GET STATUS
EF78 A880                         TEST    AL,080H                 ; MASTER READY
EF7A 750C                         JNZ     J40A                    ; TEST_DIR
EF7C E2F9                         LOOP    J39                     ; WAIT_MASTER
EF7E 800E410080                   OR      DISKETTE_STATUS,TIME_OUT
EF83                      J40:                                    ; RESULTS_ERROR
EF83 F9                           STC                             ; SET ERROR RETURN
EF84 5B                           POP     BX
EF85 5A                           POP     DX
EF86 59                           POP     CX
EF87 C3                           RET

                          ;----- TEST THE DIRECTION BIT

EF88                      J40A:
EF88 EC                           IN      AL,DX                   ; GET STATUS REG AGAIN
EF89 A840                         TEST    AL,040H                 ; TEST DIRECTION BIT
EF8B 7507                         JNZ     J42                     ; OK TO READ STATUS
EF8D                      J41:                                    ; NEC_FAIL
EF8D 800E410020                   OR      DISKETTE_STATUS,BAD_NEC
EF92 EBEF                         JMP     J40                     ; RESULTS_ERROR

                          ;----- READ IN THE STATUS

EF94                      J42:                                    ; INPUT_STAT
EF94 42                           INC     DX                      ; POINT AT DATA PORT
EF95 EC                           IN      AL,DX                   ; GET THE DATA
EF96 8805                         MOV     [DI],AL                 ; STORE THE BYTE
EF98 47                           INC     DI                      ; INCREMENT THE POINTER
EF99 B90A00                       MOV     CX,10                   ; LOOP TO KILL TIME FOR NEC
EF9C E2FE                 J43:    LOOP    J43
EF9E 4A                           DEC     DX                      ; POINT AT STATUS PORT
EF9F EC                           IN      AL,DX                   ; GET STATUS
EFA0 A810                         TEST    AL,010H                 ; TEST FOR NEC STILL BUSY
EFA2 7406                         JZ      J44                     ; RESULTS DONE
EFA4 FECB                         DEC     BL                      ; DECREMENT THE STATUS COUNTER
EFA6 75CA                         JNZ     J38                     ; GO BACK FOR MORE
EFA8 EBE3                         JMP     J41                     ; CHIP HAS FAILED

                          ;----- RESULT OPERATION IS DONE

EFAA                      J44:
EFAA 5B                           POP     BX
EFAB 5A                           POP     DX
EFAC 59                           POP     CX                      ; RECOVER REGISTERS
EFAD C3                           RET                             ; GOOD RETURN CODE FROM TEST INST
                          ;----------------------------------------------------------------
                          ; NUM_TRANS                                                      :
                          ;       THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT       :
                          ;       WERE ACTUALLY TRANSFERRED TO/FROM THE DISKETTE           :
                          ; INPUT                                                          :
                          ;       (CH) = CYLINDER OF OPERATION                             :
                          ;       (CL) = START SECTOR OF OPERATION                         :
                          ; OUTPUT                                                         :
                          ;       (AL) = NUMBER ACTUALLT TRANSFERRED                       :
                          ;       NO OTHER REGISTERS MODIFIED                              :
                          ;----------------------------------------------------------------
EFAE                      NUM_TRANS       PROC    NEAR
EFAE A04500                       MOV     AL,NEC_STATUS+3         ; GET CYLINDER ENDED UP ON
EFB1 3AC5                         CMP     AL,CH                   ; SAME AS WE STARTED
EFB3 A04700                       MOV     AL,NEC_STATUS+5         ; GET ENDING SECTOR
EFB6 740A                         JZ      J45                     ; IF ON SAME CYL, THEN NO ADJUST
EFB8 BB0800                       MOV     BX,8
EFBB E8AEFE                       CALL    GET_PARM                ; GET EOT VALUE
EFBE 8AC4                         MOV     AL,AH                   ;  INTO AL
EFC0 FEC0                         INC     AL                      ; USE EOT+1 FOR CALCULATION
EFC2                      J45:
EFC2 2AC1                         SUB     AL,CL                   ; SUBTRACT START FROM END
```

Appendix A

```
LOC OBJ              LINE    SOURCE

EFC4 C3                              RET
                             NUM_TRANS        ENDP
                             RESULTS ENDP
                             ;----------------------------------------------------------------
                             ; DISK_BASE                                                      :
                             ;        THIS IS THE SET OF PARAMETERS REQUIRED FOR OISKETTE OPERATION.  :
                             ;        THEY ARE POINTED AT BT THE OATA VARIABLE DISK_POINTER. TO      :
                             ;        NOOIFY THE PARANETERS, BUILD ANOTHER PARAMETER BLOCK AND POINT :
                             ;        OISK_POINTER TO IT.                                            :
                             ;----------------------------------------------------------------
EFC7                                 ORG     0EFC7H
EFC7                         OISK_BASE       LABEL   BTTE
EFC7 CF                              OB      11001111B       ; SRT=C, NO UNLOAD=0F - 1ST SPECIFY BYTE
EFC8 02                              OB      2               ; ND LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
EFC9 25                              DB      MOTOR_WAIT      ; WAIT AFTER OPN TIL MOTOR OFF
EFCA D2                              OB      2               ; 512 BYTES/SECTOR
EFCB 08                              OB      8               ; EOT ( LAST SECTOR ON TRACK)
EFCC 2A                              DB      02AH            ; GAP LENGTH
EFCD FF                              DB      0FFH            ; OTL
EFCE 50                              DB      050H            ; GAP LENGTH FOR FORMAT
EFCF F6                              OB      0F6H            ; FILL BTTE FOR FORMAT
EFD0 19                              DB      25              ; HEAD SETTLE TIME (MILLISECONDS)
EFD1 04                              OB      4               ; MOTOR START TIME (1/8 SECONDS)

                             ;--- INT 17 -----------------------------------------------------
                             ; PRINTER_IO                                                     :
                             ;        THIS ROUTINE PROVIOES COMMUNICATION WITH THE PRINTER      :
                             ; INPUT                                                          :
                             ;        (AH)=0  PRINT THE CHARACTER IN (AL)                     :
                             ;                ON RETURN, AH=1 IF CHARACTER COULD NOT BE PRINTED    :
                             ;                (TIME OUT). OTHER BITS SET AS ON NORMAL STATUS CALL  :
                             ;        (AH)=1  INITIALIZE THE PRINTER PORT                     :
                             ;                RETURNS WITH (AH) SET WITN PRINTER STATUS       :
                             ;        (AH)=2  READ THE PRINTER STATUS INTO (AH)               :
                             ;                7       6       5       4       3       2-1  0  :
                             ;                |       |       |       |       |       |    |_TIME OUT :
                             ;                |       |       |       |       |       |_ UNUSEO  :
                             ;                |       |       |       |       |_ 1 = I/O ERROR   :
                             ;                |       |       |       |_ 1 = SELECTEO            :
                             ;                |       |       |_ 1 = OUT OF PAPER                :
                             ;                |       |_ 1 = ACKNOWLEOGE                         :
                             ;                |_ 1 = NOT BUST                                    :
                             ;                                                                :
                             ;        (OX) = PRINTER TO BE USED (0,1,2) CORRESPONDING TO ACTUAL  :
                             ;                VALUES IN PRINTER_BASE AREA                     :
                             ;                                                                :
                             ; OATA AREA PRINTER_BASE CONTAINS THE BASE AOORESS OF THE PRINTER  :
                             ; CARO(S) AVAILABLE (LOCATEO AT BEGINNING OF DATA SEGMENT,       :
                             ; 408H ABSOLUTE, 3 WOROS)                                        :
                             ;                                                                :
                             ; OATA AREA PRINT_TIM_OUT (BTTE) MAY BE CHANGEO TO CAUSE OIFFERENT :
                             ; TIME-OUT WAITS. OEFAULT=20                                     :
                             ;                                                                :
                             ; REGISTERS      AH IS MODIFIEO                                  :
                             ;                ALL OTHERS UNCHANGED                            :
                             ;----------------------------------------------------------------
                                     ASSUME  CS:COOE,OS:OATA
EFD2                                 ORG     0EFD2H
EFD2                         PRINTER_IO      PROC    FAR
EFD2 FB                              STI                             ; INTERRUPTS BACK ON
EFD3 1E                              PUSH    DS                      ; SAVE SEGMENT
EFD4 52                              PUSH    DX
EFD5 56                              PUSH    SI
EFD6 51                              PUSH    CX
EFD7 53                              PUSH    BX
EFD8 E8630F                          CALL    DDS
EFDB 8BF2                            MOV     SI,DX                   ; GET PRINTER PARM
EFDD 8A5C78                          MOV     BL,PRINT TIM OUT[SI]    ; LOAD TIME-OUT PARM
EFE0 D1E6                            SHL     SI,1                    ; WORO OFFSET INTO TABLE
EFE2 8B5408                          MOV     DX,PRINTER_BASE[SI]     ; GET BASE ADDRESS FOR PRINTER CARD
EFE5 0BD2                            OR      DX,DX                   ; TEST DX FOR ZERO,
                                                                     ;  INDICATING NO PRINTER
EFE7 740C                            JZ      B1                      ; RETURN
EFE9 0AE4                            OR      AH,AH                   ; TEST FOR (AH)=0
EFEB 740E                            JZ      B2                      ; PRINT_AL
EFED FECC                            DEC     AH                      ; TEST FOR (AH)=1
EFEF 743F                            JZ      B8                      ; INIT_PRT
```

```
LOC OBJ          LINE     SOURCE

EFF1 FECC                         DEC     AH                 ; TEST FOR (AH)=2
EFF3 7428                         JZ      B5                 ; PRINTER STATUS
EFF5                      B1:                                ; RETURN
EFF5 5B                           POP     BX
EFF6 59                           POP     CX
EFF7 5E                           POP     SI                 ; RECOVER REGISTERS
EFF8 5A                           POP     DX                 ; RECOVER REGISTERS
EFF9 1F                           POP     DS
EFFA CF                           IRET

                          ;------ PRINT THE CHARACTER IN (AL)

EFFB                      B2:
EFFB 50                           PUSH    AX                 ; SAVE VALUE TO PRINT
EFFC EE                           OUT     DX,AL              ; OUTPUT CHAR TO PORT
EFFD 42                           INC     DX                 ; POINT TO STATUS PORT
EFFE                      B3:
EFFE 2BC9                         SUB     CX,CX              ; WAIT_BUSY
F000                      B3_1:
F000 EC                           IN      AL,DX              ; GET STATUS
F001 8AE0                         MOV     AH,AL              ; STATUS TO AH ALSO
F003 A880                         TEST    AL,80H             ; IS THE PRINTER CURRENTLY BUSY
F005 750E                         JNZ     B4                 ; OUT_STROBE
F007 E2F7                         LOOP    B3_1               ; TRY AGAIN
F009 FECB                         DEC     BL                 ; DROP LOOP COUNT
F00B 75F1                         JNZ     B3                 ; GO TILL TIMEOUT ENDS
F00D 80CC01                       OR      AH,1               ; SET ERROR FLAG
F010 80E4F9                       AND     AH,0F9H            ; TURN OFF THE OTHER BITS
F013 EB13                         JMP     SHORT B7           ; RETURN WITH ERROR FLAG SET
F015                      B4:                                ; OUT_STROBE
F015 B00D                         MOV     AL,0DH             ; SET THE STROBE HIGH
F017 42                           INC     DX                 ; STROBE IS BIT 0 OF PORT C OF 8255
F018 EE                           OUT     DX,AL
F019 B00C                         MOV     AL,0CH             ; SET THE STROBE LOW
F01B EE                           OUT     DX,AL
F01C 58                           POP     AX                 ; RECOVER THE OUTPUT CHAR

                          ;------ PRINTER STATUS

F01D                      B5:
F01D 50                           PUSH    AX                 ; SAVE AL REG
F01E                      B6:
F01E 8B5408                       MOV     DX,PRINTER_BASE[SI]
F021 42                           INC     DX
F022 EC                           IN      AL,DX              ; GET PRINTER STATUS
F023 8AE0                         MOV     AH,AL
F025 80E4F8                       AND     AH,0F8H            ; TURN OFF UNUSED BITS
F028                      B7:                                ; STATUS_SET
F028 5A                           POP     DX                 ; RECOVER AL REG
F029 8AC2                         MOV     AL,DL              ; GET CHARACTER INTO AL
F02B 80F448                       XOR     AH,48H             ; FLIP A COUPLE OF BITS
F02E EBC5                         JMP     B1                 ; RETURN FROM ROUTINE

                          ;------ INITIALIZE THE PRINTER PORT

F030                      B8:
F030 50                           PUSH    AX                 ; SAVE AL
F031 42                           INC     DX                 ; POINT TO OUTPUT PORT
F032 42                           INC     DX
F033 B008                         MOV     AL,8               ; SET INIT LINE LOW
F035 EE                           OUT     DX,AL
F036 B8E803                       MOV     AX,1000
F039                      B9:                                ; INIT_LOOP
F039 48                           DEC     AX                 ; LOOP FOR RESET TO TAKE
F03A 75FD                         JNZ     B9                 ; INIT_LOOP
F03C B00C                         MOV     AL,0CH             ; NO INTERRUPTS, NON AUTO LF,
                                                             ;  INIT HIGH
F03E EE                           OUT     DX,AL
F03F EBDD                         JMP     B6                 ; PRT_STATUS_1
                          PRINTER_IO      ENDP

F041 62E1                 C2      DW      C24                ; RETURN ADDRESS FOR DUMMY STACK

                          ;--- INT 10 -------------------------------------------------------
                          ; VIDEO_IO                                                        :
                          ;       THESE ROUTINES PROVIDE THE CRT INTERFACE                  :
                          ;       THE FOLLOWING FUNCTIONS ARE PROVIDED:                     :
```

```
;       (AH)=0  SET MODE (AL) CONTAINS MODE VALUE                       :
;               (AL)=0 40X25 BW (POWER ON DEFAULT)                      :
;               (AL)=1 40X25 COLOR                                      :
;               (AL)=2  80X25 BW                                        :
;               (AL)=3  80X25 COLOR                                     :
;               GRAPHICS MODES                                          :
;               (AL)=4  320X200 COLOR                                   :
;               (AL)=5  320X200 BW                                      :
;               (AL)=6  640X200 BW                                      :
;               CRT MODE=7 80X25 B&W CARD (USED INTERNAL TO VIDEO ONLY) :
;               *** NOTE BW MODES OPERATE SAME AS COLOR MODES, BUT      :
;                        COLOR BURST IS NOT ENABLED                     :
;       (AH)=1  SET CURSOR TYPE                                         :
;               (CH) =  BITS 4-0 = START LINE FOR CURSOR                :
;                       ** HARDWARE WILL ALWAYS CAUSE BLINK             :
;                       ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC        :
;                          BLINKING OR NO CURSOR AT ALL                 :
;               (CL) =  BITS 4-0 = END LINE FOR CURSOR                  :
;       (AH)=2  SET CURSOR POSITION                                     :
;               (DH,DL) = ROW,COLUMN  (0,0) IS UPPER LEFT               :
;               (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES)       :
;       (AH)=3  READ CURSOR POSITION                                    :
;               (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES)       :
;               ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR          :
;                       (CH,CL) = CURSOR MODE CURRENTLY SET             :
;       (AH)=4  READ LIGHT PEN POSITION                                 :
;               ON EXIT:                                                :
;               (AH) = 0 -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED     :
;               (AH) = 1 -- VALID LIGHT PEN VALUE IN REGISTERS          :
;                       (DH,DL) = ROW,COLUMN OF CHARACTER LP POSN       :
;                       (CH) = RASTER LINE (0-199)                      :
;                       (BX) = PIXEL COLUMN (0-319,639)                 :
;       (AH)=5  SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES) :
;               (AL)=NEW PAGE VAL (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3):
;       (AH)=6  SCROLL ACTIVE PAGE UP                                   :
;               (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT BOTTOM   :
;                      OF WINDOW                                        :
;                       AL = 0 MEANS BLANK ENTIRE WINDOW                :
;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL     :
;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL    :
;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE               :
;       (AH)=7  SCROLL ACTIVE PAGE DOWN                                 :
;               (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP      :
;                      OF WINDOW                                        :
;                       AL = 0 MEANS BLANK ENTIRE WINDOW                :
;               (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL     :
;               (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL    :
;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE               :
;                                                                       :
;       CHARACTER HANDLING ROUTINES                                     :
;                                                                       :
;       (AH) = 8 READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION    :
;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)        :
;               ON EXIT:                                                :
;               (AL) = CHAR READ                                        :
;               (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)   :
;       (AH) = 9 WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION   :
;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)        :
;               (CX) = COUNT OF CHARACTERS TO WRITE                     :
;               (AL) = CHAR TO WRITE                                    :
;               (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR     :
;                       (GRAPHICS)                                      :
;                       SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.      :
;       (AH) = 10 WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION       :
;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)        :
;               (CX) = COUNT OF CHARACTERS TO WRITE                     :
;               (AL) = CHAR TO WRITE                                    :
;       FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE  :
;               CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE  :
;               MAINTAINED IN THE SYSTEM ROM.  ONLY THE 1ST 128 CHARS   :
;               ARE CONTAINED THERE.  TO READ/WRITE THE SECOND 128      :
;               CHARS, THE USER MUST INITIALIZE THE POINTER AT          :
;               INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE 1K BYTE :
;               TABLE CONTAINING THE CODE POINTS FOR THE SECOND         :
;               128 CHARS (128-255).                                    :
;       FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION :
;               FACTOR CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID    :
```

```
                 ;               RESULTS ONLY FOR CHARACTERS CONTAINED ON THE SAME ROW.   :
                 ;               CONTINUATION TO SUCCEEDING LINES WILL NOT PRODUCE        :
                 ;               CORRECTLY.                                               :
                 ;                                                                        :
                 ;       GRAPHICS INTERFACE                                               :
                 ;       (AH) = 11 SET COLOR PALETTE                                      :
                 ;               (BH) = PALETTE COLOR ID BEING SET (0-127)                :
                 ;               (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID         :
                 ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT    :
                 ;                        HAS MEANING ONLY FOR 320X200 GRAPHICS.          :
                 ;                       COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15):
                 ;                       COLOR ID = 1 SELECTS THE PALETTE TO BE USED:     :
                 ;                               0 = GREEN(1)/RED(2)/YELLOW(3)            :
                 ;                               1 = CYAN(1)/MAGENTA(2)/WHITE(3)          :
                 ;                       IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET     :
                 ;                               FOR PALETTE COLOR 0 INDICATES THE        :
                 ;                               BORDER COLOR TO BE USED (VALUES 0-31,    :
                 ;                               WHERE 16-31 SELECT THE HIGH INTENSITY    :
                 ;                               BACKGROUND SET.                          :
                 ;       (AH) = 12 WRITE DOT                                              :
                 ;               (DX) = ROW NUMBER                                        :
                 ;               (CX) = COLUMN NUMBER                                     :
                 ;               (AL) = COLOR VALUE                                       :
                 ;                       IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS      :
                 ;                       EXCLUSIVE OR'D WITH THE CURRENT CONTENTS OF      :
                 ;                       THE DOT                                          :
                 ;       (AH) = 13 READ DOT                                               :
                 ;               (DX) = ROW NUMBER                                        :
                 ;               (CX) = COLUMN NUMBER                                     :
                 ;               (AL) RETURNS THE DOT READ                                :
                 ;                                                                        :
                 ; ASCII TELETYPE ROUTINE FOR OUTPUT                                      :
                 ;                                                                        :
                 ;       (AH) = 14 WRITE TELETYPE TO ACTIVE PAGE                          :
                 ;               (AL) = CHAR TO WRITE                                     :
                 ;               (BL) = FOREGROUND COLOR IN GRAPHICS MODE                 :
                 ;               NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET  :
                 ;                                                                        :
                 ;       (AH) = 15 CURRENT VIDEO STATE                                    :
                 ;               RETURNS THE CURRENT VIDEO STATE                          :
                 ;               (AL) = MODE CURRENTLY SET ( SEE AH=0 FOR EXPLANATION)    :
                 ;               (AH) = NUMBER OF CHARACTER COLUMNS ON SCREEN             :
                 ;               (BH) = CURRENT ACTIVE DISPLAY PAGE                       :
                 ;                                                                        :
                 ;       CS,SS,DS,ES,BX,CX,DX PRESERVED DURING CALL                       :
                 ;       ALL OTHERS DESTROYED                                             :
                 ;------------------------------------------------------------------------
                         ASSUME  CS:CODE,DS:DATA,ES:VIDEO_RAM
F045                     ORG     0F045H
F045             M1      LABEL   WORD                    ; TABLE OF ROUTINES WITHIN VIDEO I/O
F045 FCF0                DW      OFFSET  SET_MODE
F047 CDF1                DW      OFFSET  SET_CTYPE
F049 EEF1                DW      OFFSET  SET_CPOS
F04B 39F2                DW      OFFSET  READ_CURSOR
F04D 9CF7                DW      OFFSET  READ_LPEN
F04F 17F2                DW      OFFSET  ACT_DISP_PAGE
F051 96F2                DW      OFFSET  SCROLL_UP
F053 38F3                DW      OFFSET  SCROLL_DOWN
F055 74F3                DW      OFFSET  READ_AC_CURRENT
F057 89F3                DW      OFFSET  WRITE_AC_CURRENT
F059 ECF3                DW      OFFSET  WRITE_C_CURRENT
F05B 4EF2                DW      OFFSET  SET_COLOR
F05D 2FF4                DW      OFFSET  WRITE_DOT
F05F 1EF4                DW      OFFSET  READ_DOT
F061 18F7                DW      OFFSET  WRITE_TTY
F063 74F2                DW      OFFSET  VIDEO_STATE
  0020           M1L     EQU     $-M1

F065                     ORG     0F065H
F065             VIDEO_IO        PROC    NEAR
F065 FB                  STI                             ; INTERRUPTS BACK ON
F066 FC                  CLD                             ; SET DIRECTION FORWARD
F067 06                  PUSH    ES
F068 1E                  PUSH    DS                      ; SAVE SEGMENT REGISTERS
F069 52                  PUSH    DX
F06A 51                  PUSH    CX
F06B 53                  PUSH    BX
```

Appendix A

```
LOC OBJ                 SOURCE

F06C 56                            PUSH    SI
F06D 57                            PUSH    DI
F06E 50                            PUSH    AX                        ; SAVE AX VALUE
F06F 8AC4                          MOV     AL,AH                     ; GET INTO LOW BYTE
F071 32E4                          XOR     AH,AH                     ; ZERO TO HIGH BYTE
F073 01E0                          SAL     AX,1                      ; *2 FOR TABLE LOOKUP
F075 8BF0                          MOV     SI,AX                     ; PUT INTO SI FOR BRAHCH
F077 302000                        CMP     AX,M1L                    ; TEST FOR WITHIH RANGE
F07A 7204                          JB      M2                        ; BRANCH AROUND BRANCH
F07C 58                            POP     AX                        ; THROW AWAY THE PARAMETER
F070 E94501                        JMP     VIDEO_RETURH              ; DO HOTHIHG IF NOT IN RANGE
F080                       M2:
F080 E8BB0E                        CALL    DDS
F083 B800B8                        MOV     AX,0B800H                 ; SEGMENT FOR COLOR CARO
F086 8B3E1000                      MOV     DI,EQUIP_FLAG             ; GET EQUIPMENT SETTING
F08A 81E73000                      AND     DI,30H                    ; ISOLATE CRT SWITCHES
F08E 83FF30                        CMP     DI,30H                    ; IS SETTING FOR BW CARO?
F091 7502                          JNE     M3
F093 B4B0                          MOV     AH,0B0H                   ; SEGMEHT FOR BW CARD
F095                       M3:
F095 8EC0                          MOV     ES,AX                     ; SET UP TO POINT AT VIDEO RAM AREAS
F097 58                            POP     AX                        ; RECOVER VALUE
F098 8A264900                      MOV     AH,CRT_MODE               ; GET CURRENT MODE INTD AH
F09C 2EFFA445F0                    JMP     WORO PTR CS:[SI+OFFSET M1]
                           VIDEO_IO        ENDP
                           ;--------------------------------------------------------
                           ; SET_MODE                                              :
                           ;          THIS ROUTIHE INITIALIZES THE ATTACHMENT TO   :
                           ;          THE SELECTED MODE.  THE SCREEN IS BLANKED.   :
                           ; INPUT                                                 :
                           ;          (AL) = MODE SELECTED (RANGE 0-9)             :
                           ; OUTPUT                                                :
                           ;          NONE                                         :
                           ;--------------------------------------------------------

                           ;----- TABLES FOR USE IH SETTING OF MODE

F0A4                               ORG     0F0A4H
F0A4                       VIDEO_PARMS     LABEL   BYTE
                           ;----- INIT_TABLE
F0A4 38                            DB      38H,28H,2DH,0AH,1FH,6,19H        ; SET UP FOR 40X25
F0A5 28
F0A6 2D
F0A7 0A
F0A8 1F
F0A9 06
F0AA 19
F0AB 1C                            DB      1CH,2,7,6,7
F0AC 02
F0AD 07
F0AE 06
F0AF 07
F0B0 00                            DB      0,0,0,0
F0B1 00
F0B2 00
F0B3 00
  0010                     M4      EQU     $-VIDEO_PARMS

F0B4 71                            DB      71H,50H,5AH,0AH,1FH,6,19H        ; SET UP FOR 80X25
F0B5 50
F0B6 5A
F0B7 0A
F0B8 1F
F0B9 06
F0BA 19
F0BB 1C                            DB      1CH,2,7,6,7
F0BC 02
F0BD 07
F0BE 06
F0BF 07
F0C0 00                            DB      0,0,0,0
F0C1 00
F0C2 00
F0C3 00

F0C4 38                            DB      38H,28H,2DH,0AH,7FH,6,64H        ; SET UP FOR GRAPHICS
F0C5 28
```

```
LOC OBJ         SOURCE

F0C6 2D
F0C7 0A
F0C8 7F
F0C9 06
F0CA 64
F0CB 70                 DB      70H,2,1,6,7
F0CC 02
F0CD 01
F0CE 06
F0CF 07
F0D0 00                 DB      0,0,0,0
F0D1 00
F0D2 00
F0D3 00
F0D4 61                 DB      61H,50H,52H,0FH,19H,6,19H       ; SET UP FOR 80X25 B&W CARD
F0D5 50
F0D6 52
F0D7 0F
F0D8 19
F0D9 06
F0DA 19
F0DB 19                 DB      19H,2,0DH,0BH,0CH
F0DC 02
F0DD 0D
F0DE 0B
F0DF 0C
F0E0 00                 DB      0,0,0,0
F0E1 00
F0E2 00
F0E3 00
F0E4            M5      LABEL   WORD                    ; TABLE OF REGEN LENGTHS
F0E4 0008               DW      2048                    ; 40X25
F0E6 0010               DW      4096                    ; 80X25
F0E8 0040               DW      16384                   ; GRAPHICS
F0EA 0040               DW      16384
                ;----- COLUMNS
F0EC            M6      LABEL   BYTE
F0EC 28                 DB      40,40,80,80,40,40,80,80
F0ED 28
F0EE 50
F0EF 50
F0F0 28
F0F1 28
F0F2 50
F0F3 50
                ;----- C_REG_TAB
F0F4            M7      LABEL   BYTE                    ; TABLE OF MODE SETS
F0F4 2C                 DB      2CH,28H,2DH,29H,2AH,2EH,1EH,29H
F0F5 28
F0F6 2D
F0F7 29
F0F8 2A
F0F9 2E
F0FA 1E
F0FB 29
F0FC            SET_MODE        PROC    NEAR
F0FC BAD403             MOV     DX,03D4H                ; ADDRESS OF COLOR CARD
F0FF B300               MOV     BL,0                    ; MODE SET FOR COLOR CARD
F101 83FF30             CMP     DI,30H                  ; IS BW CARD INSTALLED
F104 7506               JNE     M8                      ; OK WITH COLOR
F106 B007               MOV     AL,7                    ; INDICATE BW CARD MODE
F108 B2B4               MOV     DL,0B4H                 ; ADDRESS OF BW CARD (3B4)
F10A FEC3               INC     BL                      ; MODE SET FOR BW CARD
F10C            M8:
F10C 8AE0               MOV     AH,AL                   ; SAVE MODE IN AH
F10E A24900             MOV     CRT_MODE,AL             ; SAVE IN GLOBAL VARIABLE
F111 89166300           MOV     ADDR_6845,DX            ; SAVE ADDRESS OF BASE
F115 1E                 PUSH    DS                      ; SAVE POINTER TO DATA SEGMENT
F116 50                 PUSH    AX                      ; SAVE MODE
F117 52                 PUSH    DX                      ; SAVE OUTPUT PORT VALUE
```

Appendix A

```
LOC OBJ                 SOURCE

F118 83C204                     ADD     DX,4                    ; POINT TO CONTROL REGISTER
F11B 8AC3                       MOV     AL,BL                   ; GET MODE SET FOR CARD
F11D EE                         OUT     DX,AL                   ; RESET VIDEO
F11E 5A                         POP     DX                      ; BACK TO BASE REGISTER
F11F 2BC0                       SUB     AX,AX                   ; SET UP FOR ABS0 SEGMENT
F121 8ED8                       MOV     DS,AX                   ; ESTABLISH VECTOR TABLE ADDRESSING
                                ASSUME  DS:ABS0
F123 C51E7400                   LDS     BX,PARM_PTR             ; GET POINTER TO VIDEO PARMS
F127 58                         POP     AX                      ; RECOVER PARMS
                                ASSUME  DS:CODE
F128 B91000                     MOV     CX,M4                   ; LENGTH OF EACH ROW OF TABLE
F12B 80FC02                     CMP     AH,2                    ; DETERMINE WHICH ONE TO USE
F12E 7210                       JC      M9                      ; MODE IS 0 OR 1
F130 03D9                       ADD     BX,CX                   ; MOVE TO NEXT ROW OF INIT TABLE
F132 80FC04                     CMP     AH,4
F135 7209                       JC      M9                      ; MODE IS 2 OR 3
F137 03D9                       ADD     BX,CX                   ; MOVE TO GRAPHICS ROW OF INIT_TABLE
F139 80FC07                     CMP     AH,7
F13C 7202                       JC      M9                      ; MODE IS 4,5, OR 6
F13E 03D9                       ADD     BX,CX                   ; MOVE TO BW CARD ROW OF INIT_TABLE

                        ;----- BX POINTS TO CORRECT ROW OF INITIALIZATION TABLE

F140                    M9:                                     ; OUT_INIT
F140 50                         PUSH    AX                      ; SAVE MODE IN AH
F141 32E4                       XOR     AH,AH                   ; AH WILL SERVE AS REGISTER
                                                                ;  NUMBER DURING LOOP

                        ;----- LOOP THROUGH TABLE, OUTPUTTING REG ADDRESS, THEN VALUE FROM TABLE

F143                    M10:                                    ; INIT LOOP
F143 8AC4                       MOV     AL,AH                   ; GET 6845 REGISTER NUMBER
F145 EE                         OUT     DX,AL
F146 42                         INC     DX                      ; POINT TO DATA PORT
F147 FEC4                       INC     AH                      ; NEXT REGISTER VALUE
F149 8A07                       MOV     AL,[BX]                 ; GET TABLE VALUE
F14B EE                         OUT     DX,AL                   ; OUT TO CHIP
F14C 43                         INC     BX                      ; NEXT IN TABLE
F14D 4A                         DEC     DX                      ; BACK TO POINTER REGISTER
F14E E2F3                       LOOP    M10                     ; DO THE WHOLE TABLE
F150 58                         POP     AX                      ; GET MODE BACK
F151 1F                         POP     DS                      ; RECOVER SEGMENT VALUE
                                ASSUME  DS:DATA

                        ;----- FILL REGEN AREA WITH BLANK

F152 33FF                       XOR     DI,DI                   ; SET UP POINTER FOR REGEN
F154 893E4E00                   MOV     CRT_START,DI            ; START ADDRESS SAVED IN GLOBAL
F158 C606620000                 MOV     ACTIVE_PAGE,0           ; SET PAGE VALUE
F15D B90020                     MOV     CX,8192                 ; NUMBER OF WORDS IN COLOR CARD
F160 80FC04                     CMP     AH,4                    ; TEST FOR GRAPHICS
F163 720B                       JC      M12                     ; NO_GRAPHICS_INIT
F165 80FC07                     CMP     AH,7                    ; TEST FOR BW CARD
F168 7404                       JE      M11                     ; BW_CARD_INIT
F16A 33C0                       XOR     AX,AX                   ; FILL FOR GRAPHICS MODE
F16C EB05                       JMP     SHORT M13               ; CLEAR_BUFFER
F16E                    M11:                                    ; BW_CARD_INIT
F16E B508                       MOV     CH,08H                  ; BUFFER SIZE ON BW CARD
F170                    M12:                                    ; NO_GRAPHICS_INIT
F170 B82007                     MOV     AX,' '+7*256            ; FILL CHAR FOR ALPHA
F173                    M13:                                    ; CLEAR_BUFFER
F173 F3                         REP     STOSW                   ; FILL THE REGEN BUFFER WITH BLANKS
F174 AB

                        ;----- ENABLE VIDEO AND CORRECT PORT SETTING

F175 C70660000706               MOV     CURSOR_MODE,607H        ; SET CURRENT CURSOR MODE
F17B A04900                     MOV     AL,CRT_MODE             ; GET THE MODE
F17E 32E4                       XOR     AH,AH                   ;  INTO AX REGISTER
F180 8BF0                       MOV     SI,AX                   ; TABLE POINTER, INDEXED BY MODE
F182 8B166300                   MOV     DX,ADDR_6845            ; PREPARE TO OUTPUT TO
                                                                ;  VIDEO ENABLE PORT
F186 83C204                     ADD     DX,4
F189 2E8A84F4F0                 MOV     AL,CS:[SI+OFFSET M7]
F18E EE                         OUT     DX,AL                   ; SET VIDEO ENABLE PORT
F18F A26500                     MOV     CRT_MODE_SET,AL         ; SAVE THAT VALUE
```

```
LOC OBJ                 SOURCE

                        ;----- DETERMINE NUMBER OF COLUMNS, BOTH FOR ENTIRE DISPLAY
                        ;----- AND THE NUMBER TO BE USED FOR TTY INTERFACE

F192 2E8A84ECF0                 MOV     AL,CS:[SI + OFFSET M6]
F197 32E4                       XOR     AH,AH
F199 A34A00                     MOV     CRT_COLS,AX             ; NUMBER OF COLUMNS IN THIS SCREEN

                        ;----- SET CURSOR POSITIONS

F19C 81E60E00                   AND     SI,0EH                  ; WORD OFFSET INTO CLEAR LENGTH TABLE
F1A0 2E8B8CE4F0                 MOV     CX,CS:[SI + OFFSET M5]  ; LENGTH TO CLEAR
F1A5 890E4C00                   MOV     CRT_LEN,CX              ; SAVE LENGTH OF CRT -- NOT USED FOR BW
F1A9 B90800                     MOV     CX,8                    ; CLEAR ALL CURSOR POSITIONS
F1AC BF5000                     MOV     DI,OFFSET CURSOR_POSN
F1AF 1E                         PUSH    DS                      ; ESTABLISH SEGMENT
F1B0 07                         POP     ES                      ;   ADDRESSING
F1B1 33C0                       XOR     AX,AX
F1B3 F3                         REP     STOSW                   ; FILL WITH ZEROES
F1B4 AB

                        ;----- SET UP OVERSCAN REGISTER

F1B5 42                         INC     DX                      ; SET OVERSCAN PORT TO A DEFAULT
F1B6 B030                       MOV     AL,30H                  ; VALUE OF 30H FOR ALL MODES
                                                                ;  EXCEPT 640X200
F1B8 803E490006                 CMP     CRT_MODE,6              ; SEE IF THE MODE IS 640X200 BW
F1BD 7502                       JNZ     M14                     ; IF IT ISNT 640X200, THEN GOTO REGULAR
F1BF B03F                       MOV     AL,3FH                  ; IF IT IS 640X200, THEN PUT IN 3FH
F1C1                    M14:
F1C1 EE                         OUT     DX,AL                   ; OUTPUT THE CORRECT VALUE TO 3D9 PORT
F1C2 A26600                     MOV     CRT_PALETTE,AL          ; SAVE THE VALUE FOR FUTURE USE

                        ;----- NORMAL RETURN FROM ALL VIDEO RETURNS

F1C5                    VIDEO_RETURN:
F1C5 5F                         POP     DI
F1C6 5E                         POP     SI
F1C7 5B                         POP     BX
F1C8                    M15:                                    ; VIDEO_RETURN_C
F1C8 59                         POP     CX
F1C9 5A                         POP     DX
F1CA 1F                         POP     DS
F1CB 07                         POP     ES                      ; RECOVER SEGMENTS
F1CC CF                         IRET                            ; ALL DONE
                        SET_MODE        ENDP
                        ;--------------------------------------------------------------
                        ; SET_CTYPE                                                   :
                        ;       THIS ROUTINE SETS THE CURSOR VALUE                    :
                        ; INPUT                                                       :
                        ;       (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE     :
                        ; OUTPUT                                                      :
                        ;       NONE                                                  :
                        ;--------------------------------------------------------------
F1CD                    SET_CTYPE       PROC    NEAR
F1CD B40A                       MOV     AH,10                   ; 6845 REGISTER FOR CURSOR SET
F1CF 890E6000                   MOV     CURSOR_MODE,CX          ; SAVE IN DATA AREA
F1D3 E80200                     CALL    M16                     ; OUTPUT CX REG
F1D6 EBED                       JMP     VIDEO_RETURN

                        ;----- THIS ROUTINE OUTPUTS THE CX REGISTER TO THE 6845 REGS NAMED IN AH

F1D8                    M16:
F1D8 8B166300                   MOV     DX,ADDR_6845            ; ADDRESS REGISTER
F1DC 8AC4                       MOV     AL,AH                   ; GET VALUE
F1DE EE                         OUT     DX,AL                   ; REGISTER SET
F1DF 42                         INC     DX                      ; DATA REGISTER
F1E0 8AC5                       MOV     AL,CH                   ; DATA
F1E2 EE                         OUT     DX,AL
F1E3 4A                         DEC     DX
F1E4 8AC4                       MOV     AL,AH
F1E6 FEC0                       INC     AL                      ; POINT TO OTHER DATA REGISTER
F1E8 EE                         OUT     DX,AL                   ; SET FOR SECOND REGISTER
F1E9 42                         INC     DX
F1EA 8AC1                       MOV     AL,CL                   ; SECOND DATA VALUE
F1EC EE                         OUT     DX,AL
F1ED C3                         RET                             ; ALL DONE
                        SET_CTYPE       ENDP
```

Appendix A

```
LOC OBJ                 SOURCE

                        ;--------------------------------------------------
                        ; SET_CPOS                                        :
                        ;       THIS ROUTINE SETS THE CURRENT CURSOR      :
                        ;       POSITION TO THE NEW X-Y VALUES PASSED     :
                        ; INPUT                                           :
                        ;       DX - ROW,COLUMN OF NEW CURSOR             :
                        ;       BH - DISPLAY PAGE OF CURSOR               :
                        ; OUTPUT                                          :
                        ;       CURSOR IS SET AT 6845 IF DISPLAY PAGE     :
                        ;       IS CURRENT DISPLAY                        :
                        ;--------------------------------------------------
F1EE                    SET_CPOS        PROC    NEAR
F1EE 8ACF                       MOV     CL,BH
F1F0 32ED                       XOR     CH,CH                   ; ESTABLISH LOOP COUNT
F1F2 D1E1                       SAL     CX,1                    ; WORD OFFSET
F1F4 8BF1                       MOV     SI,CX                   ; USE INDEX REGISTER
F1F6 895450                     MOV     [SI+OFFSET CURSOR_POSN],DX      ; SAVE THE POINTER
F1F9 383E6200                   CMP     ACTIVE_PAGE,BH
F1FD 7505                       JNZ     M17                     ; SET_CPOS_RETURN
F1FF 8BC2                       MOV     AX,DX                   ; GET ROW/COLUMN TO AX
F201 E80200                     CALL    M18                     ; CURSOR_SET
F204                    M17:                                    ; SET_CPOS_RETURN
F204 EBBF                       JMP     VIDEO_RETURN
                        SET_CPOS        ENDP

                        ;----- SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR

F206                    M18     PROC    NEAR
F206 E87C00                     CALL    POSITION                ; DETERMINE LOCATION IN REGEN BUFFER
F209 8BC8                       MOV     CX,AX
F20B 030E4E00                   ADD     CX,CRT_START            ; ADD IN THE START ADDR FOR THIS PAGE
F20F D1F9                       SAR     CX,1                    ; DIVIDE BY 2 FOR CHAR ONLY COUNT
F211 B40E                       MOV     AH,14                   ; REGISTER NUMBER FOR CURSOR
F213 E8C2FF                     CALL    M16                     ; OUTPUT THE VALUE TO THE 6845
F216 C3                         RET
                        M18     ENDP
                        ;-------------------------------------------------------------
                        ; ACT_DISP_PAGE                                              :
                        ;       THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING THE :
                        ;       FULL USE OF THE RAM SET ASIDE FOR THE VIDEO ATTACHMENT  :
                        ; INPUT                                                      :
                        ;       AL HAS THE NEW ACTIVE DISPLAY PAGE                   :
                        ; OUTPUT                                                     :
                        ;       THE 6845 IS RESET TO DISPLAY THAT PAGE               :
                        ;-------------------------------------------------------------
F217                    ACT_DISP_PAGE   PROC    NEAR
F217 A26200                     MOV     ACTIVE_PAGE,AL          ; SAVE ACTIVE PAGE VALUE
F21A 8B0E4C00                   MOV     CX,CRT_LEN              ; GET SAVED LENGTH OF REGEN BUFFER
F21E 98                         CBW                             ; CONVERT AL TO WORD
F21F 50                         PUSH    AX                      ; SAVE PAGE VALUE
F220 F7E1                       MUL     CX                      ; DISPLAY PAGE TIMES REGEN LENGTH
F222 A34E00                     MOV     CRT_START,AX            ; SAVE START ADDRESS FOR
                                                                ;  LATER REQUIREMENTS
F225 8BC8                       MOV     CX,AX                   ; START ADDRESS TO CX
F227 D1F9                       SAR     CX,1                    ; DIVIDE BY 2 FOR 6845 HANDLING
F229 B40C                       MOV     AH,12                   ; 6845 REGISTER FOR START ADDRESS
F22B E8AAFF                     CALL    M16
F22E 5B                         POP     BX                      ; RECOVER PAGE VALUE
F22F D1E3                       SAL     BX,1                    ; *2 FOR WORD OFFSET
F231 8B4750                     MOV     AX,[BX + OFFSET CURSOR_POSN]    ; GET CURSOR FOR THIS PAGE
F234 E8CFFF                     CALL    M18                     ; SET THE CURSOR POSITION
F237 EB8C                       JMP     SHORT VIDEO_RETURN
                        ACT_DISP_PAGE   ENDP
                        ;-------------------------------------------------------------
                        ; READ_CURSOR                                                :
                        ;       THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE :
                        ;       6845, FORMATS IT, AND SENDS IT BACK TO THE CALLER    :
                        ; INPUT                                                      :
                        ;       BH - PAGE OF CURSOR                                  :
                        ; OUTPUT                                                     :
                        ;       DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION      :
                        ;       CX - CURRENT CURSOR MODE                             :
                        ;-------------------------------------------------------------
F239                    READ_CURSOR     PROC    NEAR
F239 8ADF                       MOV     BL,BH
F23B 32FF                       XOR     BH,BH
F23D D1E3                       SAL     BX,1                    ; WORD OFFSET
```

```
LOC OBJ                 LINE     SOURCE

F23F 8B5750                      MOV     DX,[BX+OFFSET CURSOR_POSN]
F242 8B0E6000                    MOV     CX,CURSOR_MODE
F246 5F                          POP     DI
F247 5E                          POP     SI
F248 5B                          POP     BX
F249 58                          POP     AX                      ; DISCARD SAVED CX AND DX
F24A 58                          POP     AX
F24B 1F                          POP     DS
F24C 07                          POP     ES
F24D CF                          IRET
                         READ_CURSOR     ENDP
                         ;--------------------------------------------------------------------
                         ; SET COLOR                                                         :
                         ;       THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN  :
                         ;       COLOR, AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION      :
                         ;       GRAPHICS                                                    :
                         ; INPUT                                                             :
                         ;       (BH) HAS COLOR ID                                           :
                         ;               IF BH=0, THE BACKGROUND COLOR VALUE IS SET          :
                         ;                       FROM THE LOW BITS OF BL (0-31)              :
                         ;               IF BH=1, THE PALETTE SELECTION IS MADE              :
                         ;                       BASED ON THE LOW BIT OF BL:                 :
                         ;                               0=GREEN, RED, YELLOW FOR COLORS 1,2,3  :
                         ;                               1=BLUE, CYAN, MAGENTA FOR COLORS 1,2,3 :
                         ;       (BL) HAS THE COLOR VALUE TO BE USED                         :
                         ; OUTPUT                                                            :
                         ;       THE COLOR SELECTION IS UPDATED                              :
                         ;--------------------------------------------------------------------
F24E                     SET_COLOR       PROC    NEAR
F24E 8B166300                    MOV     DX,ADDR_6845            ; I/O PORT FOR PALETTE
F252 83C205                      ADD     DX,5                    ; OVERSCAN PORT
F255 A06600                      MOV     AL,CRT_PALETTE          ; GET THE CURRENT PALETTE VALUE
F258 0AFF                        OR      BH,BH                   ; IS THIS COLOR 0?
F25A 750E                        JNZ     M20                     ; OUTPUT COLOR 1

                         ;----- HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR

F25C 24E0                        AND     AL,0E0H                 ; TURN OFF LOW 5 BITS OF CURRENT
F25E 80E31F                      AND     BL,01FH                 ; TURN OFF HIGH 3 BITS OF INPUT VALUE
F261 0AC3                        OR      AL,BL                   ; PUT VALUE INTO REGISTER
F263                     M19:                                    ; OUTPUT THE PALETTE
F263 EE                          OUT     DX,AL                   ; OUTPUT COLOR SELECTION TO 3D9 PORT
F264 A26600                      MOV     CRT_PALETTE,AL          ; SAVE THE COLOR VALUE
F267 E95BFF                      JMP     VIDEO_RETURN

                         ;----- HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED

F26A                     M20:
F26A 240F                        AND     AL,0DFH                 ; TURN OFF PALETTE SELECT BIT
F26C D0EB                        SHR     BL,1                    ; TEST THE LOW ORDER BIT OF BL
F26E 73F3                        JNC     M19                     ; ALREADY DONE
F270 0C20                        OR      AL,20H                  ; TURN ON PALETTE SELECT BIT
F272 EBEF                        JMP     M19                     ; GO DO IT
                         SET_COLOR       ENDP
                         ;----------------------------------------------
                         ; VIDEO STATE                                 :
                         ;  RETURNS THE CURRENT VIDEO STATE IN AX      :
                         ;  AH = NUMBER OF COLUMNS ON THE SCREEN       :
                         ;  AL = CURRENT VIDEO MODE                    :
                         ;  BH = CURRENT ACTIVE PAGE                   :
                         ;----------------------------------------------
F274                     VIDEO_STATE     PROC    NEAR
F274 8A264A00                    MOV     AH,BYTE PTR CRT_COLS    ; GET NUMBER OF COLUMNS
F278 A04900                      MOV     AL,CRT_MODE             ; CURRENT MODE
F27B 8A3E6200                    MOV     BH,ACTIVE_PAGE          ; GET CURRENT ACTIVE PAGE
F27F 5F                          POP     DI                      ; RECOVER REGISTERS
F280 5E                          POP     SI
F281 59                          POP     CX                      ; DISCARD SAVED BX
F282 E943FF                      JMP     M15                     ; RETURN TO CALLER
                         VIDEO_STATE     ENDP
                         ;------------------------------------------------------
                         ; POSITION                                            :
                         ;       THIS SERVICE ROUTINE CALCULATES THE REGEN      :
                         ;       BUFFER ADDRESS OF A CHARACTER IN THE ALPHA MODE :
                         ; INPUT                                               :
                         ;       AX = ROW, COLUMN POSITION                     :
                         ; OUTPUT                                              :
```

Appendix A

```
LOC OBJ                 SOURCE

                        ;       AX = OFFSET OF CHAR POSITION IN REGEN BUFFER    :
                        ;--------------------------------------------------------
F285                    POSITION        PROC    NEAR
F285 53                         PUSH    BX                      ; SAVE REGISTER
F286 8BD8                       MOV     BX,AX
F288 8AC4                       MOV     AL,AH                   ; ROWS TO AL
F28A F6264A00                   MUL     BYTE PTR CRT_COLS       ; DETERMINE BYTES TO ROW
F28E 32FF                       XOR     BH,BH
F290 03C3                       ADD     AX,BX                   ; ADD IN COLUMN VALUE
F292 D1E0                       SAL     AX,1                    ; * 2 FOR ATTRIBUTE BYTES
F294 5B                         POP     BX
F295 C3                         RET
                        POSITION        ENDP
                        ;--------------------------------------------------------
                        ; SCROLL UP                                             :
                        ;       THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP     :
                        ;       ON THE SCREEN                                   :
                        ; INPUT                                                 :
                        ;       (AH) = CURRENT CRT MODE                         :
                        ;       (AL) = NUMBER OF ROWS TO SCROLL                 :
                        ;       (CX) = ROW/COLUMN OF UPPER LEFT CORNER          :
                        ;       (DX) = ROW/COLUMN OF LOWER RIGHT CORNER         :
                        ;       (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE     :
                        ;       (DS) = DATA SEGMENT                             :
                        ;       (ES) = REGEN BUFFER SEGMENT                     :
                        ; OUTPUT                                                :
                        ;       NONE -- THE REGEN BUFFER IS MODIFIED            :
                        ;--------------------------------------------------------
                                ASSUME  CS:CODE,DS:DATA,ES:DATA
F296                    SCROLL_UP       PROC    NEAR
F296 8AD8                       MOV     BL,AL                   ; SAVE LINE COUNT IN BL
F298 80FC04                     CMP     AH,4                    ; TEST FOR GRAPHICS MODE
F29B 7208                       JC      N1                      ; HANDLE SEPARATELY
F29D 80FC07                     CMP     AH,7                    ; TEST FOR BW CARD
F2A0 7403                       JE      N1
F2A2 E9F001                     JMP     GRAPHICS_UP
F2A5                    N1:                                     ; UP_CONTINUE
F2A5 53                         PUSH    BX                      ; SAVE FILL ATTRIBUTE IN BH
F2A6 8BC1                       MOV     AX,CX                   ; UPPER LEFT POSITION
F2A8 E83700                     CALL    SCROLL_POSITION         ; DO SETUP FOR SCROLL
F2AB 7431                       JZ      N7                      ; BLANK_FIELD
F2AD 03F0                       ADD     SI,AX                   ; FROM ADDRESS
F2AF 8AE6                       MOV     AH,DH                   ; # ROWS IN BLOCK
F2B1 2AE3                       SUB     AH,BL                   ; # ROWS TO BE MOVED
F2B3                    N2:                                     ; ROW_LOOP
F2B3 E87200                     CALL    N10                     ; MOVE ONE ROW
F2B6 03F5                       ADD     SI,BP
F2B8 03FD                       ADD     DI,BP                   ; POINT TO NEXT LINE IN BLOCK
F2BA FECC                       DEC     AH                      ; COUNT OF LINES TO MOVE
F2BC 75F5                       JNZ     N2                      ; ROW_LOOP
F2BE                    N3:                                     ; CLEAR_ENTRY
F2BE 58                         POP     AX                      ; RECOVER ATTRIBUTE IN AH
F2BF B020                       MOV     AL,' '                  ; FILL WITH BLANKS
F2C1                    N4:                                     ; CLEAR_LOOP
F2C1 E86D00                     CALL    N11                     ; CLEAR THE ROW
F2C4 03FD                       ADD     DI,BP                   ; POINT TO NEXT LINE
F2C6 FECB                       DEC     BL                      ; COUNTER OF LINES TO SCROLL
F2C8 75F7                       JNZ     N4                      ; CLEAR_LOOP
F2CA                    N5:                                     ; SCROLL_END
F2CA E8710C                     CALL    DDS
F2CD 803E490007                 CMP     CRT_MODE,7              ; IS THIS THE BLACK AND WHITE CARD
F2D2 7407                       JE      N6                      ; IF SO, SKIP THE MODE RESET
F2D4 A06500                     MOV     AL,CRT_MODE_SET         ; GET THE VALUE OF THE MODE SET
F2D7 BAD803                     MOV     DX,03D8H                ; ALWAYS SET COLOR CARD PORT
F2DA EE                         OUT     DX,AL
F2DB                    N6:                                     ; VIDEO_RET_HERE
F2DB E9E7FE                     JMP     VIDEO_RETURN
F2DE                    N7:                                     ; BLANK_FIELD
F2DE 8ADE                       MOV     BL,DH                   ; GET ROW COUNT
F2E0 EBDC                       JMP     N3                      ; GO CLEAR THAT AREA
                        SCROLL_UP       ENDP

                        ;----- HANDLE COMMON SCROLL SET UP HERE

F2E2                    SCROLL_POSITION PROC    NEAR
F2E2 803E490002                 CMP     CRT_MODE,2              ; TEST FOR SPECIAL CASE HERE
F2E7 7218                       JB      N9                      ; HAVE TO HANDLE 80X25 SEPARATELY
```

```
F2E9 803E490003          CMP     CRT_MODE,3
F2EE 7711                JA      N9

                   ;----- 80X25 COLOR CARD SCROLL

F2F0 52                  PUSH    DX
F2F1 BADA03              MOV     DX,3DAH                 ; GUARANTEED TO BE COLOR CARD HERE
F2F4 50                  PUSH    AX
F2F5               N8:                                   ; WAIT_DISP_ENABLE
F2F5 EC                  IN      AL,DX                   ; GET PORT
F2F6 A808                TEST    AL,8                    ; WAIT FOR VERTICAL RETRACE
F2F8 74FB                JZ      N8                      ; WAIT_DISP_ENABLE
F2FA B025                MOV     AL,25H
F2FC B2D8                MOV     DL,0D8H                 ; DX=3D8
F2FE EE                  OUT     DX,AL                   ; TURN OFF VIDEO
F2FF 58                  POP     AX                      ; DURING VERTICAL RETRACE
F300 5A                  POP     DX
F301               N9:
F301 E881FF              CALL    POSITION                ; CONVERT TO REGEN POINTER
F304 03064E00            ADD     AX,CRT_START            ; OFFSET OF ACTIVE PAGE
F308 8BF8                MOV     DI,AX                   ; TO ADDRESS FOR SCROLL
F30A 8BF0                MOV     SI,AX                   ; FROM ADDRESS FOR SCROLL
F30C 2BD1                SUB     DX,CX                   ; DX = #ROWS, #COLS IN BLOCK
F30E FEC6                INC     DH
F310 FEC2                INC     DL                      ; INCREMENT FOR 0 ORIGIN
F312 32ED                XOR     CH,CH                   ; SET HIGH BYTE OF COUNT TO ZERO
F314 8B2E4A00            MOV     BP,CRT_COLS             ; GET NUMBER OF COLUMNS IN DISPLAY
F318 03ED                ADD     BP,BP                   ; TIMES 2 FOR ATTRIBUTE BYTE
F31A 8AC3                MOV     AL,BL                   ; GET LINE COUNT
F31C F6264A00            MUL     BYTE PTR CRT_COLS       ; DETERMINE OFFSET TO FROM ADDRESS
F320 03C0                ADD     AX,AX                   ; *2 FOR ATTRIBUTE BYTE
F322 06                  PUSH    ES                      ; ESTABLISH ADDRESSING TO REGEN BUFFER
F323 1F                  POP     DS                      ;  FOR BOTH POINTERS
F324 80FB00              CMP     BL,0                    ; 0 SCROLL MEANS BLANK FIELD
F327 C3                  RET                             ; RETURN WITH FLAGS SET
                   SCROLL_POSITION ENDP

                   ;----- MOVE_ROW

F328               N10   PROC    NEAR
F328 8ACA                MOV     CL,DL                   ; GET # OF COLS TO MOVE
F32A 56                  PUSH    SI
F32B 57                  PUSH    DI                      ; SAVE START ADDRESS
F32C F3                  REP     MOVSW                   ; MOVE THAT LINE ON SCREEN
F32D A5
F32E 5F                  POP     DI
F32F 5E                  POP     SI                      ; RECOVER ADDRESSES
F330 C3                  RET
                   N10   ENDP

                   ;----- CLEAR_ROW

F331               N11   PROC    NEAR
F331 8ACA                MOV     CL,DL                   ; GET # COLUMNS TO CLEAR
F333 57                  PUSH    DI
F334 F3                  REP     STOSW                   ; STORE THE FILL CHARACTER
F335 AB
F336 5F                  POP     DI
F337 C3                  RET
                   N11   ENDP
                   ;---------------------------------------------------------
                   ; SCROLL_DOWN                                            :
                   ;       THIS ROUTINE MOVES THE CHARACTERS WITHIN A       :
                   ;       DEFINED BLOCK DOWN ON THE SCREEN, FILLING THE    :
                   ;       TOP LINES WITH A DEFINED CHARACTER               :
                   ; INPUT                                                  :
                   ;       (AH) = CURRENT CRT MODE                          :
                   ;       (AL) = NUMBER OF LINES TO SCROLL                 :
                   ;       (CX) = UPPER LEFT CORNER OF REGION               :
                   ;       (DX) = LOWER RIGHT CORNER OF REGION              :
                   ;       (BH) = FILL CHARACTER                            :
                   ;       (DS) = DATA SEGMENT                              :
                   ;       (ES) = REGEN SEGMENT                             :
                   ; OUTPUT                                                 :
                   ;       NONE -- SCREEN IS SCROLLED                       :
                   ;---------------------------------------------------------
F338               SCROLL_DOWN     PROC    NEAR
```

```
LOC OBJ                 SOURCE

F338 FD                          STD                        ; DIRECTION FOR SCROLL DOWN
F339 8AD8                        MOV     BL,AL              ; LINE COUNT TO BL
F33B 80FC04                      CMP     AH,4               ; TEST FOR GRAPHICS
F33E 7208                        JC      N12
F340 80FC07                      CMP     AH,7               ; TEST FOR BW CARD
F343 7403                        JE      N12
F345 E9A601                      JMP     GRAPHICS_DOWN
F348                     N12:                               ; CONTINUE DOWN
F348 53                          PUSH    BX                 ; SAVE ATTRIBUTE IN BH
F349 8BC2                        MOV     AX,DX              ; LOWER RIGHT CORNER
F34B E894FF                      CALL    SCROLL_POSITION    ; GET REGEN LOCATION
F34E 7420                        JZ      N16
F350 2BF0                        SUB     SI,AX              ; SI IS FROM ADDRESS
F352 8AE6                        MOV     AH,DH              ; GET TOTAL # ROWS
F354 2AE3                        SUB     AH,BL              ; COUNT TO MOVE IN SCROLL
F356                     N13:
F356 E8CFFF                      CALL    N10                ; MOVE ONE ROW
F359 2BF5                        SUB     SI,BP
F35B 2BFD                        SUB     DI,BP
F35D FECC                        DEC     AH
F35F 75F5                        JNZ     N13
F361                     N14:
F361 58                          POP     AX                 ; RECOVER ATTRIBUTE IN AH
F362 B020                        MOV     AL,' '
F364                     N15:
F364 E8CAFF                      CALL    N11                ; CLEAR ONE ROW
F367 2BFD                        SUB     DI,BP              ; GO TO NEXT ROW
F369 FECB                        DEC     BL
F36B 75F7                        JNZ     N15
F36D E95AFF                      JMP     N5                 ; SCROLL_END
F370                     N16:
F370 8ADE                        MOV     BL,DH
F372 EBE0                        JMP     N14
                         SCROLL_DOWN     ENDP
                         ;--------------------------------------------------------
                         ; READ_AC_CURRENT                                        :
                         ;       THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER   :
                         ;       AT THE CURRENT CURSOR POSITION AND RETURNS THEM  :
                         ;       TO THE CALLER                                    :
                         ;INPUT                                                  :
                         ;       (AH) = CURRENT CRT MODE                          :
                         ;       (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )         :
                         ;       (DS) = DATA SEGMENT                              :
                         ;       (ES) = REGEN SEGMENT                             :
                         ;OUTPUT                                                 :
                         ;       (AL) = CHAR READ                                 :
                         ;       (AH) = ATTRIBUTE READ                            :
                         ;--------------------------------------------------------
                                 ASSUME  CS:CODE,DS:DATA,ES:DATA
F374                     READ_AC_CURRENT PROC    NEAR
F374 80FC04                      CMP     AH,4               ; IS THIS GRAPHICS
F377 7208                        JC      P1
F379 80FC07                      CMP     AH,7               ; IS THIS BW CARD
F37C 7403                        JE      P1
F37E E9A802                      JMP     GRAPHICS_READ
F381                     P1:                                ; READ_AC_CONTINUE
F381 E81A00                      CALL    FIND_POSITION
F384 8BF3                        MOV     SI,BX              ; ESTABLISH ADDRESSING IN SI

                         ;----- WAIT FOR HORIZONTAL RETRACE

F386 8B166300                    MOV     DX,ADDR_6845       ; GET BASE ADDRESS
F38A 83C206                      ADD     DX,6               ; POINT AT STATUS PORT
F38D 06                          PUSH    ES
F38E 1F                          POP     DS                 ; GET SEGMENT FOR QUICK ACCESS
F38F                     P2:                                ; WAIT FOR RETRACE LOW
F38F EC                          IN      AL,DX              ; GET STATUS
F390 A801                        TEST    AL,1               ; IS HORZ RETRACE LOW
F392 75FB                        JNZ     P2                 ; WAIT UNTIL IT IS
F394 FA                          CLI                        ; NO MORE INTERRUPTS
F395                     P3:                                ; WAIT FOR RETRACE HIGH
F395 EC                          IN      AL,DX              ; GET STATUS
F396 A801                        TEST    AL,1               ; IS IT HIGH
F398 74FB                        JZ      P3                 ; WAIT UNTIL IT IS
F39A AD                          LODSW                      ; GET THE CHAR/ATTR
F39B E927FE                      JMP     VIDEO_RETURN
                         READ_AC_CURRENT ENDP
```

```
LOC OBJ          LINE    SOURCE

F39E                     FIND_POSITION   PROC    NEAR
F39E 8ACF                        MOV     CL,BH                   ; DISPLAY PAGE TO CX
F3A0 32ED                        XOR     CH,CH
F3A2 8BF1                        MOV     SI,CX                   ; MOVE TO SI FOR INDEX
F3A4 D1E6                        SAL     SI,1                    ; * 2 FOR WORD OFFSET
F3A6 8B4450                      MOV     AX,[SI+ OFFSET CURSOR_POSN]     ; GET ROW/COLUMN OF THAT PAGE
F3A9 33DB                        XOR     BX,BX                   ; SET START ADDRESS TO ZERO
F3AB E306                        JCXZ    P5                      ; NO_PAGE
F3AD                     P4:                                     ; PAGE_LOOP
F3AD 031E4C00                    ADD     BX,CRT_LEN              ; LENGTH OF BUFFER
F3B1 E2FA                        LOOP    P4
F3B3                     P5:                                     ; NO_PAGE
F3B3 E8CFFE                      CALL    POSITION                ; DETERMINE LOCATION IN REGEN
F3B6 03D8                        ADD     BX,AX                   ; ADD TO START OF REGEN
F3B8 C3                          RET
                         FIND_POSITION   ENDP
                         ;------------------------------------------------
                         ; WRITE_AC_CURRENT                              :
                         ;       THIS ROUTINE WRITES THE ATTRIBUTE       :
                         ;       AND CHARACTER AT THE CURRENT CURSOR     :
                         ;       POSITION                                :
                         ; INPUT                                         :
                         ;       (AH) = CURRENT CRT MODE                 :
                         ;       (BH) = DISPLAY PAGE                     :
                         ;       (CX) = COUNT OF CHARACTERS TO WRITE     :
                         ;       (AL) = CHAR TO WRITE                    :
                         ;       (BL) = ATTRIBUTE OF CHAR TO WRITE       :
                         ;       (DS) = DATA SEGMENT                     :
                         ;       (ES) = REGEN SEGMENT                    :
                         ; OUTPUT                                        :
                         ;       NONE                                    :
                         ;------------------------------------------------
F3B9                     WRITE_AC_CURRENT        PROC    NEAR
F3B9 80FC04                      CMP     AH,4                    ; IS THIS GRAPHICS
F3BC 7208                        JC      P6
F3BE 80FC07                      CMP     AH,7                    ; IS THIS BW CARD
F3C1 7403                        JE      P6
F3C3 E9B201                      JMP     GRAPHICS_WRITE
F3C6                     P6:                                     ; WRITE_AC_CONTINUE
F3C6 8AE3                        MOV     AH,BL                   ; GET ATTRIBUTE TO AH
F3C8 50                          PUSH    AX                      ; SAVE ON STACK
F3C9 51                          PUSH    CX                      ; SAVE WRITE COUNT
F3CA E8D1FF                      CALL    FIND_POSITION
F3CD 8BFB                        MOV     DI,BX                   ; ADDRESS TO DI REGISTER
F3CF 59                          POP     CX                      ; WRITE COUNT
F3D0 5B                          POP     BX                      ; CHARACTER IN BX REG
F3D1                     P7:                                     ; WRITE_LOOP

                         ;----- WAIT FOR HORIZONTAL RETRACE

F3D1 8B166300                    MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F3D5 83C206                      ADD     DX,6                    ; POINT AT STATUS PORT
F3D8                     P8:
F3D8 EC                          IN      AL,DX                   ; GET STATUS
F3D9 A801                        TEST    AL,1                    ; IS IT LOW
F3DB 75FB                        JNZ     P8                      ; WAIT UNTIL IT IS
F3DD FA                          CLI                             ; NO MORE INTERRUPTS
F3DE                     P9:
F3DE EC                          IN      AL,DX                   ; GET STATUS
F3DF A801                        TEST    AL,1                    ; IS IT HIGH
F3E1 74FB                        JZ      P9                      ; WAIT UNTIL IT IS
F3E3 8BC3                        MOV     AX,BX                   ; RECOVER THE CHAR/ATTR
F3E5 AB                          STOSW                           ; PUT THE CHAR/ATTR
F3E6 FB                          STI                             ; INTERRUPTS BACK ON
F3E7 E2E8                        LOOP    P7                      ;  AS MANY TIMES AS REQUESTED
F3E9 E9D9FD                      JMP     VIDEO_RETURN
                         WRITE_AC_CURRENT        ENDP
                         ;------------------------------------------------
                         ; WRITE_C_CURRENT                               :
                         ;       THIS ROUTINE WRITES THE CHARACTER AT    :
                         ;       THE CURRENT CURSOR POSITION, ATTRIBUTE  :
                         ;       UNCHANGED                               :
                         ; INPUT                                         :
                         ;       (AH) = CURRENT CRT MODE                 :
                         ;       (BH) = DISPLAY PAGE                     :
                         ;       (CX) = COUNT OF CHARACTERS TO WRITE     :
```

Appendix A

```
LOC OBJ                 SOURCE

                        ;       (AL) = CHAR TO WRITE                  :
                        ;       (DS) = DATA SEGMENT                   :
                        ;       (ES) = REGEN SEGMENT                  :
                        ; OUTPUT                                      :
                        ;       NONE                                  :
                        ;----------------------------------------------
F3EC                    WRITE_C_CURRENT PROC    NEAR
F3EC 80FC04                     CMP     AH,4                    ; IS THIS GRAPHICS
F3EF 7208                       JC      P10
F3F1 80FC07                     CMP     AH,7                    ; IS THIS BW CARD
F3F4 7403                       JE      P10
F3F6 E97F01                     JMP     GRAPHICS_WRITE
F3F9                    P10:
F3F9 50                         PUSH    AX                      ; SAVE ON STACK
F3FA 51                         PUSH    CX                      ; SAVE WRITE COUNT
F3FB E8A0FF                     CALL    FIND_POSITION
F3FE 8BFB                       MOV     DI,BX                   ; ADDRESS TO DI
F400 59                         POP     CX                      ; WRITE COUNT
F401 5B                         POP     BX                      ; BL HAS CHAR TO WRITE
F402                    P11:                                    ; WRITE_LOOP

                        ;----- WAIT FOR HORIZONTAL RETRACE

F402 8B166300                   MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F406 83C206                     ADD     DX,6                    ; POINT AT STATUS PORT
F409                    P12:
F409 EC                         IN      AL,DX                   ; GET STATUS
F40A A801                       TEST    AL,1                    ; IS IT LOW
F40C 75FB                       JNZ     P12                     ; WAIT UNTIL IT IS
F40E FA                         CLI                             ; NO MORE INTERRUPTS
F40F                    P13:
F40F EC                         IN      AL,DX                   ; GET STATUS
F410 A801                       TEST    AL,1                    ; IS IT HIGH
F412 74FB                       JZ      P13                     ; WAIT UNTIL IT IS
F414 8AC3                       MOV     AL,BL                   ; RECOVER CHAR
F416 AA                         STOSB                           ; PUT THE CHAR/ATTR
F417 FB                         STI                             ; INTERRUPTS BACK ON
F418 47                         INC     DI                      ; BUMP POINTER PAST ATTRIBUTE
F419 E2E7                       LOOP    P11                     ;  AS MANY TIMES AS REQUESTED
F41B E9A7FD                     JMP     VIDEO_RETURN
                        WRITE_C_CURRENT ENDP
                        ;--------------------------------------------------------------
                        ; READ DOT  -- WRITE DOT                                      :
                        ;       THESE ROUTINES WILL WRITE A DOT, OR READ THE DOT AT   :
                        ;       THE INDICATED LOCATION                                :
                        ; ENTRY --                                                    :
                        ;   DX = ROW (0-199)    (THE ACTUAL VALUE DEPENDS ON THE MODE) :
                        ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )  :
                        ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,  :
                        ;       REQ'D FOR WRITE DOT ONLY, RIGHT JUSTIFIED)            :
                        ;       BIT 7 OF AL=1 INDICATES XOR THE VALUE INTO THE LOCATION :
                        ;   DS = DATA SEGMENT                                         :
                        ;   ES = REGEN SEGMENT                                        :
                        ;                                                             :
                        ; EXIT                                                        :
                        ;       AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY       :
                        ;--------------------------------------------------------------
                                ASSUME  CS:CODE,DS:DATA,ES:DATA
F41E                    READ_DOT        PROC    NEAR
F41E E83100                     CALL    R3                      ; DETERMINE BYTE POSITION OF DOT
F421 268A04                     MOV     AL,ES:[SI]              ; GET THE BYTE
F424 22C4                       AND     AL,AH                   ; MASK OFF THE OTHER BITS IN THE BYTE
F426 D2E0                       SHL     AL,CL                   ; LEFT JUSTIFY THE VALUE
F428 8ACE                       MOV     CL,DH                   ; GET NUMBER OF BITS IN RESULT
F42A D2C0                       ROL     AL,CL                   ; RIGHT JUSTIFY THE RESULT
F42C E996FD                     JMP     VIDEO_RETURN            ; RETURN FROM VIDEO IO
                        READ_DOT        ENDP

F42F                    WRITE_DOT       PROC    NEAR
F42F 50                         PUSH    AX                      ; SAVE DOT VALUE
F430 50                         PUSH    AX                      ;  TWICE
F431 E81E00                     CALL    R3                      ; DETERMINE BYTE POSITION OF THE DOT
F434 D2E8                       SHR     AL,CL                   ; SHIFT TO SET UP THE BITS FOR OUTPUT
F436 22C4                       AND     AL,AH                   ; STRIP OFF THE OTHER BITS
F438 268A0C                     MOV     CL,ES:[SI]              ; GET THE CURRENT BYTE
F43B 5B                         POP     BX                      ; RECOVER XOR FLAG
F43C F6C380                     TEST    BL,80H                  ; IS IT ON
```

```
LOC OBJ            SOURCE

F43F 750D                JNZ     R2                     ; YES, XOR THE DOT
F441 F6D4                NOT     AH                     ; SET THE MASK TO REMOVE THE
F443 22CC                AND     CL,AH                  ;  INDICATED BITS
F445 0AC1                OR      AL,CL                  ; OR IN THE NEW VALUE OF THOSE BITS
F447               R1:                                  ; FINISH_DOT
F447 268804              MOV     ES:[SI],AL             ; RESTORE THE BYTE IN MEMORY
F44A 58                  POP     AX
F44B E977FD              JMP     VIDEO_RETURN           ; RETURN FROM VIDEO IO
F44E               R2:                                  ; XOR_DOT
F44E 32C1                XOR     AL,CL                  ; EXCLUSIVE OR THE DOTS
F450 E8F5                JMP     R1                     ; FINISH UP THE WRITING
                   WRITE_DOT     ENDP
                   ;-------------------------------------------------------
                   ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION   :
                   ; OF THE INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.  :
                   ; ENTRY --                                            :
                   ;  DX = ROW VALUE (0-199)                             :
                   ;  CX = COLUMN VALUE (0-639)                          :
                   ; EXIT --                                             :
                   ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST  :
                   ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST        :
                   ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH :
                   ;  DH = # BITS IN RESULT                              :
                   ;-------------------------------------------------------
F452               R3    PROC    NEAR
F452 53                  PUSH    BX                     ; SAVE BX DURING OPERATION
F453 50                  PUSH    AX                     ; WILL SAVE AL DURING OPERATION

                   ;----- DETERMINE 1ST BYTE IN IDICATED ROW BY MULTIPLYING ROW VALUE BY 40
                   ;----- ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW

F454 B028                MOV     AL,40
F456 52                  PUSH    DX                     ; SAVE ROW VALUE
F457 80E2FE              AND     DL,0FEH                ; STRIP OFF ODD/EVEN BIT
F45A F6E2                MUL     DL                     ; AX HAS ADDRESS OF 1ST BYTE
                                                        ;  OF INDICATED ROW
F45C 5A                  POP     DX                     ; RECOVER IT
F45D F6C201              TEST    DL,1                   ; TEST FOR EVEN/ODD
F460 7403                JZ      R4                     ; JUMP IF EVEN ROW
F462 050020              ADD     AX,2000H               ; OFFSET TO LOCATION OF ODD ROWS
F465               R4:                                  ; EVEN_ROW
F465 8BF0                MOV     SI,AX                  ; MOVE POINTER TO SI
F467 58                  POP     AX                     ; RECOVER AL VALUE
F468 8BD1                MOV     DX,CX                  ; COLUMN VALUE TO DX

                   ;----- DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT

                   ;-----------------------------------------------------------------
                   ; SET UP THE REGISTERS ACCORDING TO THE MODE                     :
                   ;  CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES)   :
                   ;  CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M)         :
                   ;  BL = MASK TO SELECT BITS FROM POINTED BYTE (80H/C0H FOR H/M) :
                   ;  BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M)      :
                   ;-----------------------------------------------------------------

F46A BBC002              MOV     BX,2C0H
F46D B90203              MOV     CX,302H                ; SET PARMS FOR MED RES
F470 803E490006          CMP     CRT_MODE,6
F475 7206                JC      R5                     ; HANDLE IF MED ARES
F477 BB8001              MOV     BX,180H
F47A B90307              MOV     CX,703H                ; SET PARMS FOR HIGH RES

                   ;----- DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK

F47D               R5:
F47D 22EA                AND     CH,DL                  ; ADDRESS OF PEL WITHIN BYTE TO CH

                   ;----- DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN

F47F D3EA                SHR     DX,CL                  ; SHIFT BY CORRECT AMOUNT
F481 03F2                ADD     SI,DX                  ; INCREMENT THE POINTER
F483 8AF7                MOV     DH,BH                  ; GET THE # OF BITS IN RESULT TO DH

                   ;----- MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)

F485 2AC9                SUB     CL,CL                  ; ZERO INTO STORAGE LOCATION
F487               R6:
```

Appendix A

```
LOC OBJ                 LINE    SOURCE

F487 D0C8                               ROR     AL,1                    ; LEFT JUSTIFY THE VALUE
                                                                        ;  IN AL (FOR WRITE)
F489 02C0                               ADD     CL,CH                   ; ADD IN THE BIT OFFSET VALUE
F48B FECF                               DEC     BH                      ; LOOP CONTROL
F48D 75F8                               JNZ     R6                      ; ON EXIT, CL HAS SHIFT COUNT
                                                                        ;  TO RESTORE BITS
F48F 8AE3                               MOV     AH,BL                   ; GET MASK TO AH
F491 D2EC                               SHR     AH,CL                   ; MOVE THE MASK TO CORRECT LOCATION
F493 5B                                 POP     BX                      ; RECOVER REG
F494 C3                                 RET                             ; RETURN WITH EVERYTHING SET UP
                                R3      ENDP
                                ;---------------------------------------------------------------
                                ; SCROLL UP                                                     :
                                ;       THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT      :
                                ; ENTRY                                                         :
                                ;       CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL           :
                                ;       DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL          :
                                ;        BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS           :
                                ;       BH = FILL VALUE FOR BLANKED LINES                       :
                                ;       AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE     :
                                ;            FIELD)                                             :
                                ;       DS = DATA SEGMENT                                       :
                                ;       ES = REGEN SEGMENT                                      :
                                ; EXIT                                                          :
                                ;       NOTHING, THE SCREEN IS SCROLLED                         :
                                ;---------------------------------------------------------------
F495                            GRAPHICS_UP     PROC    NEAR
F495 8AD8                               MOV     BL,AL                   ; SAVE LINE COUNT IN BL
F497 8BC1                               MOV     AX,CX                   ; GET UPPER LEFT POSITION INTO AX REG

                                ;----- USE CHARACTER SUBROUTINE FOR POSITIONING
                                ;----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE

F499 E86902                             CALL    GRAPH_POSN
F49C 8BF8                               MOV     DI,AX                   ; SAVE RESULT AS DESTINATION ADDRESS

                                ;----- DETERMINE SIZE OF WINDOW

F49E 2BD1                               SUB     DX,CX
F4A0 81C20101                           ADD     DX,101H                 ; ADJUST VALUES
F4A4 D0E6                               SAL     DH,1                    ; MULTIPLY # ROWS BY 4
                                                                        ;  SINCE 8 VERT DOTS/CHAR
F4A6 D0E6                               SAL     DH,1                    ;  AND EVEN/ODD ROWS

                                ;----- DETERMINE CRT MODE

F4A8 803E490006                         CMP     CRT_MODE,6              ; TEST FOR MEDIUM RES
F4AD 7304                               JNC     R7                      ; FIND_SOURCE

                                ;----- MEDIUM RES UP

F4AF D0E2                               SAL     DL,1                    ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
F4B1 D1E7                               SAL     DI,1                    ; OFFSET *2 SINCE 2 BYTES/CHAR

                                ;----- DETERMINE THE SOURCE ADDRESS IN THE BUFFER

F4B3                            R7:                                     ; FIND_SOURCE
F4B3 06                                 PUSH    ES                      ; GET SEGMENTS BOTH POINTING TO REGEN
F4B4 1F                                 POP     DS
F4B5 2AED                               SUB     CH,CH                   ; ZERO TO HIGH OF COUNT REG
F4B7 D0E3                               SAL     BL,1                    ; MULTIPLY NUMBER OF LINES BY 4
F4B9 D0E3                               SAL     BL,1
F4BB 742D                               JZ      R11                     ; IF ZERO, THEN BLANK ENTIRE FIELD
F4BD 8AC3                               MOV     AL,BL                   ; GET NUMBER OF LINES IN AL
F4BF B450                               MOV     AH,80                   ; 80 BYTES/ROW
F4C1 F6E4                               MUL     AH                      ; DETERMINE OFFSET TO SOURCE
F4C3 8BF7                               MOV     SI,DI                   ; SET UP SOURCE
F4C5 03F0                               ADD     SI,AX                   ;  ADD IN OFFSET TO IT
F4C7 8AE6                               MOV     AH,DH                   ; NUMBER OF ROWS IN FIELD
F4C9 2AE3                               SUB     AH,BL                   ; DETERMINE NUMBER TO MOVE

                                ;----- LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS

F4CB                            R8:                                     ; ROW_LOOP
F4CB E88000                             CALL    R17                     ; MOVE ONE ROW
F4CE 81EEB01F                           SUB     SI,2000H-80             ; MOVE TO NEXT ROW
F4D2 81EFB01F                           SUB     DI,2000H-80
```

```
LOC OBJ              SOURCE

F406 FECC                    DEC     AH                  ; NUMBER OF ROWS TO MOVE
F408 75F1                    JNZ     R8                  ; CONTINUE TILL ALL MOVED

                     ;----- FILL IN THE VACATED LINE(S)

F40A                 R9:                                 ; CLEAR_ENTRY
F40A 8AC7                    MOV     AL,BH               ; ATTRIBUTE TO FILL WITH
F40C                 R10:
F40C E88800                  CALL    R18                 ; CLEAR THAT ROW
F40F 81EFB01F                SUB     DI,2000H-80         ; POINT TO NEXT LINE
F4E3 FECB                    DEC     BL                  ; NUMBER OF LINES TO FILL
F4E5 75F5                    JNZ     R10                 ; CLEAR_LOOP
F4E7 E90BFC                  JMP     VIDEO_RETURN        ; EVERYTHING DONE
F4EA                 R11:                                ; BLANK_FIELD
F4EA 8ADE                    MOV     BL,DH               ; SET BLANK COUNT TO
                                                         ;  EVERYTHING IN FIELD
F4EC EBEC                    JMP     R9                  ; CLEAR THE FIELD
                     GRAPHICS_UP     ENDP
                     ;-------------------------------------------------------------
                     ; SCROLL DOWN                                                 :
                     ;       THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT  :
                     ; ENTRY                                                       :
                     ;       CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL         :
                     ;       DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL        :
                     ;        BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS         :
                     ;       BH = FILL VALUE FOR BLANKED LINES                     :
                     ;       AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE   :
                     ;            FIELD)                                           :
                     ;       DS = DATA SEGMENT                                     :
                     ;       ES = REGEN SEGMENT                                    :
                     ; EXIT                                                        :
                     ;       NOTHING, THE SCREEN IS SCROLLED                       :
                     ;-------------------------------------------------------------
F4EE                 GRAPHICS_DOWN   PROC    NEAR
F4EE FD                      STD                         ; SET DIRECTION
F4EF 8AD8                    MOV     BL,AL               ; SAVE LINE COUNT IN BL
F4F1 8BC2                    MOV     AX,DX               ; GET LOWER RIGHT POSITION INTO AX REG

                     ;----- USE CHARACTER SUBROUTINE FOR POSITIONING
                     ;----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE

F4F3 E80F02                  CALL    GRAPH_POSN
F4F6 8BF8                    MOV     DI,AX               ; SAVE RESULT AS DESTINATION ADDRESS

                     ;----- DETERMINE SIZE OF WINDOW

F4F8 2BD1                    SUB     DX,CX
F4FA 81C20101                ADD     DX,101H             ; ADJUST VALUES
F4FE D0E6                    SAL     DH,1                ; MULTIPLY # ROWS BY 4
                                                         ;  SINCE 8 VERT DOTS/CHAR
F500 D0E6                    SAL     DH,1                ;  AND EVEN/ODD ROWS

                     ;----- DETERMINE CRT MODE

F502 803E490006              CMP     CRT_MODE,6          ; TEST FOR MEDIUM RES
F507 7305                    JNC     R12                 ; FIND_SOURCE_DOWN

                     ;----- MEDIUM RES DOWN

F509 D0E2                    SAL     DL,1                ; # COLUMNS * 2, SINCE
                                                         ;  2 BYTES/CHAR (OFFSET OK)
F50B D1E7                    SAL     DI,1                ; OFFSET *2 SINCE 2 BYTES/CHAR
F50D 47                      INC     DI                  ; POINT TO LAST BYTE

                     ;----- DETERMINE THE SOURCE ADDRESS IN THE BUFFER

F50E                 R12:                                ; FIND_SOURCE_DOWN
F50E 06                      PUSH    ES                  ; BOTH SEGMENTS TO REGEN
F50F 1F                      POP     DS
F510 2AED                    SUB     CH,CH               ; ZERO TO HIGH OF COUNT REG
F512 81C7F000                ADD     DI,240              ; POINT TO LAST ROW OF PIXELS
F516 D0E3                    SAL     BL,1                ; MULTIPLY NUMBER OF LINES BY 4
F518 D0E3                    SAL     BL,1
F51A 742E                    JZ      R16                 ; IF ZERO, THEN BLANK ENTIRE FIELD
F51C 8AC3                    MOV     AL,BL               ; GET NUMBER OF LINES IN AL
F51E B450                    MOV     AH,80               ; 80 BYTES/ROW
F520 F6E4                    MUL     AH                  ; DETERMINE OFFSET TO SOURCE
```

Appendix A

```
LOC OBJ             LINE    SOURCE

F522 8BF7                       MOV     SI,OI                   ; SET UP SOURCE
F524 2BF0                       SUB     5I,AX                   ;  SUBTRACT THE OFF5ET
F526 8AE6                       MOV     AH,DH                   ; NUMBER OF ROWS IN FIELD
F528 2AE3                       SUB     AH,BL                   ; DETERMINE NUMBER TO MOVE

                        ;----- LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS

F52A                    R13:                                    ; ROW_LOOP_DOWN
F52A E82100                     CALL    R17                     ; MOVE ONE ROW
F52D 81EE5020                   SUB     SI,2000H+80             ; MOVE TO NEXT ROW
F531 81EF5020                   SUB     OI,2000H+80
F535 FECC                       DEC     AH                      ; NUMBER OF ROW5 TO MOVE
F537 75F1                       JNZ     R13                     ; CONTINUE TILL ALL MOVED

                        ;----- FILL IN THE VACATED LINE(S)

F539                    R14:                                    ; CLEAR_ENTRY_DOWN
F539 8AC7                       MOV     AL,BH                   ; ATTRIBUTE TO FILL WITH
F53B                    R15:                                    ; CLEAR_LOOP_DOWN
F53B E82900                     CALL    R18                     ; CLEAR A ROW
F53E 81EF5020                   SUB     DI,2000H+80             ; POINT TO NEXT LINE
F542 FECB                       DEC     BL                      ; NUMBER OF LINES TO FILL
F544 75F5                       JNZ     R15                     ; CLEAR_LOOP_DOWN
F546 FC                         CLD                             ; RESET THE DIRECTION FLAG
F547 E97BFC                     JMP     VIDEO_RETURN            ; EVERYTHING DONE
F54A                    R16:                                    ; BLANK_FIELD_DOWN
F54A 8ADE                       MOV     BL,DH                   ; 5ET BLANK COUNT TO
                                                                ;  EVERYTHING IN FIELD
F54C EBEB                       JMP     R14                     ; CLEAR THE FIELD
                        GRAPHICS_DOWN   ENDP

                        ;----- ROUTINE TO MOVE ONE ROW OF INFORMATION

F54E                    R17     PROC    NEAR
F54E 8ACA                       MOV     CL,OL                   ; NUMBER OF BYTES IN THE ROW
F550 56                         PUSH    SI
F551 57                         PUSH    OI                      ; SAVE POINTERS
F552 F3                         REP     MOV5B                   ; MOVE THE EVEN FIELD
F553 A4
F554 5F                         POP     DI
F555 5E                         POP     SI
F556 81C60020                   ADD     SI,2000H
F55A 81C70020                   ADD     OI,2000H                ; POINT TO THE ODD FIELD
F55E 56                         PUSH    5I
F55F 57                         PUSH    OI                      ; SAVE THE POINTERS
F560 8ACA                       MOV     CL,OL                   ; COUNT BACK
F562 F3                         REP     MOVSB                   ; MOVE THE ODD FIELD
F563 A4
F564 5F                         POP     DI
F565 5E                         POP     SI                      ; POINTERS BACK
F566 C3                         RET                             ; RETURN TO CALLER
                        R17     ENDP

                        ;----- CLEAR A SINGLE ROW

F567                    R18     PROC    NEAR
F567 8ACA                       MOV     CL,DL                   ; NUMBER OF BYTES IN FIELD
F569 57                         PUSH    DI                      ; SAVE POINTER
F56A F3                         REP     STOSB                   ; STORE THE NEW VALUE
F56B AA
F56C 5F                         POP     DI                      ; POINTER BACK
F56D 81C70020                   ADD     DI,2000H                ; POINT TO ODD FIELD
F571 57                         PUSH    OI
F572 8ACA                       MOV     CL,DL
F574 F3                         REP     STOSB                   ; FILL THE ODD FILELD
F575 AA
F576 5F                         POP     DI
F577 C3                         RET                             ; RETURN TO CALLER
                        R18     ENDP
                        ;---------------------------------------------------------
                        ; GRAPHICS WRITE                                          :
                        ;       THIS ROUTINE WRITES THE ASCII CHARACTER TO THE    :
                        ;       CURRENT POSITION ON THE SCREEN.                   :
                        ; ENTRY                                                   :
                        ;       AL = CHARACTER TO WRITE                           :
                        ;       BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR :
                        ;        IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN :
```

```
                ;         BUFFER (0 IS USED FOR THE BACKGROUND COLOR)          :
                ;        CX = NUMBER OF CHARS TO WRITE                         :
                ;        DS = DATA SEGMENT                                     :
                ;        ES = REGEN SEGMENT                                    :
                ; EXIT                                                         :
                ;        NOTHING IS RETURNED                                   :
                ;                                                              :
                ; GRAPHICS READ                                                :
                ;        THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT :
                ;        CURSOR POSITION ON THE SCREEN BY MATCHING THE DOTS ON :
                ;        THE SCREEN TO THE CHARACTER GENERATOR CODE POINTS     :
                ; ENTRY                                                        :
                ;        NONE  ( 0 IS ASSUMED AS THE BACKGROUND COLOR          :
                ; EXIT                                                         :
                ;        AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF   :
                ;             NONE FOUND)                                      :
                ;                                                              :
                ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE         :
                ;  CONTAINED IN ROM FOR THE 1ST 128 CHARS.  TO ACCESS CHARS    :
                ;  IN THE SECOND HALF, THE USER MUST INITIALIZE THE VECTOR AT  :
                ;  INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE USER        :
                ;  SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).               :
                ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS              :
                ;--------------------------------------------------------------
                        ASSUME  CS:CODE,DS:DATA,ES:DATA
F578            GRAPHICS_WRITE  PROC    NEAR
F578 B400               MOV     AH,0                    ; ZERO TO HIGH OF CODE POINT
F57A 50                 PUSH    AX                      ; SAVE CODE POINT VALUE

                ;----- DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS

F57B E88401             CALL    S26                     ; FIND LOCATION IN REGEN BUFFER
F57E 8BF8               MOV     DI,AX                   ; REGEN POINTER IN DI

                ;----- DETERMINE REGION TO GET CODE POINTS FROM

F580 58                 POP     AX                      ; RECOVER CODE POINT
F581 3C80               CMP     AL,80H                  ; IS IT IN SECOND HALF
F583 7306               JAE     S1                      ; YES

                ;----- IMAGE IS IN FIRST HALF, CONTAINED IN ROM

F585 BE6EFA             MOV     SI,0FA6EH               ; CRT_CHAR_GEN (OFFSET OF IMAGES)
F588 0E                 PUSH    CS                      ; SAVE SEGMENT ON STACK
F589 EB0F               JMP     SHORT S2                ; DETERMINE_MODE

                ;----- IMAGE IS IN SECOND HALF, IN USER RAM

F58B            S1:                                     ; EXTEND_CHAR
F58B 2C80               SUB     AL,80H                  ; ZERO ORIGIN FOR SECOND HALF
F58D 1E                 PUSH    DS                      ; SAVE DATA POINTER
F58E 2BF6               SUB     SI,SI
F590 8EDE               MOV     DS,SI                   ; ESTABLISH VECTOR ADDRESSING
                        ASSUME  DS:ABS0
F592 C5367C00           LDS     SI,EXT_PTR              ; GET THE OFFSET OF THE TABLE
F596 8CDA               MOV     DX,DS                   ; GET THE SEGMENT OF THE TABLE
                        ASSUME  DS:DATA
F598 1F                 POP     DS                      ; RECOVER DATA SEGMENT
F599 52                 PUSH    DX                      ; SAVE TABLE SEGMENT ON STACK

                ;----- DETERMINE GRAPHICS MODE IN OPERATION

F59A            S2:                                     ; DETERMINE_MODE
F59A D1E0               SAL     AX,1                    ; MULTIPLY CODE POINT
F59C D1E0               SAL     AX,1                    ;  VALUE BY 8
F59E D1E0               SAL     AX,1
F5A0 03F0               ADD     SI,AX                   ; SI HAS OFFSET OF DESIRED CODES
F5A2 803E490006         CMP     CRT_MODE,6
F5A7 1F                 POP     DS                      ; RECOVER TABLE POINTER SEGMENT
F5A8 722C               JC      S7                      ; TEST FOR MEDIUM RESOLUTION MODE

                ;----- HIGH RESOLUTION MODE

F5AA            S3:                                     ; HIGH_CHAR
F5AA 57                 PUSH    DI                      ; SAVE REGEN POINTER
F5AB 56                 PUSH    SI                      ; SAVE CODE POINTER
F5AC B604               MOV     DH,4                    ; NUMBER OF TIMES THROUGH LOOP
```

Appendix A

```
LOC OBJ                 SOURCE

F5AE                    S4:
F5AE AC                         LODSB                           ; GET BYTE FROM CODE POINTS
F5AF F6C380                     TEST    BL,80H                  ; SHOULD WE USE THE FUNCTION
F5B2 7516                       JNZ     S6                      ;  TO PUT CHAR IN
F5B4 AA                         STOSB                           ; STORE IN REGEN BUFFER
F5B5 AC                         LODSB
F5B6                    S5:
F5B6 268885FF1F                 MOV     ES:[DI+2000H-1],AL      ; STORE IN SECOND HALF
F5BB 83C74F                     ADD     DI,79                   ; MOVE TO NEXT ROW IN REGEN
F5BE FECE                       DEC     DH                      ; DONE WITH LOOP
F5C0 75EC                       JNZ     S4
F5C2 5E                         POP     SI
F5C3 5F                         POP     DI                      ; RECOVER REGEN POINTER
F5C4 47                         INC     DI                      ; POINT TO NEXT CHAR POSITION
F5C5 E2E3                       LOOP    S3                      ; MORE CHARS TO WRITE
F5C7 E9FBFB                     JMP     VIDEO_RETURN
F5CA                    S6:
F5CA 263205                     XOR     AL,ES:[DI]              ; EXCLUSIVE OR WITH CURRENT
F5CD AA                         STOSB                           ; STORE THE CODE POINT
F5CE AC                         LODSB                           ; AGAIN FOR ODD FIELD
F5CF 263285FF1F                 XOR     AL,ES:[DI+2000H-1]
F5D4 EBE0                       JMP     S5                      ; BACK TO MAINSTREAM

                        ;----- MEDIUM RESOLUTION WRITE

F5D6                    S7:                                     ; MED_RES_WRITE
F5D6 8AD3                       MOV     DL,BL                   ; SAVE HIGH COLOR BIT
F5D8 D1E7                       SAL     DI,1                    ; OFFSET*2 SINCE 2 BYTES/CHAR
F5DA E8D100                     CALL    S19                     ; EXPAND BL TO FULL WORD OF COLOR
F5DD                    S8:                                     ; MED_CHAR
F5DD 57                         PUSH    DI                      ; SAVE REGEN POINTER
F5DE 56                         PUSH    SI                      ; SAVE THE CODE POINTER
F5DF B604                       MOV     DH,4                    ; NUMBER OF LOOPS
F5E1                    S9:
F5E1 AC                         LODSB                           ; GET CODE POINT
F5E2 E80E00                     CALL    S21                     ; DOUBLE UP ALL THE BITS
F5E5 23C3                       AND     AX,BX                   ; CONVERT THEM TO FOREGROUND
                                                                ;  COLOR ( 0 BACK )
F5E7 F6C280                     TEST    DL,80H                  ; IS THIS XOR FUNCTION
F5EA 7407                       JZ      S10                     ; NO, STORE IT IN AS IT IS
F5EC 263225                     XOR     AH,ES:[DI]              ; DO FUNCTION WITH HALF
F5EF 26324501                   XOR     AL,ES:[DI+1]            ;  AND WITH OTHER HALF
F5F3                    S10:
F5F3 268825                     MOV     ES:[DI],AH              ; STORE FIRST BYTE
F5F6 26884501                   MOV     ES:[DI+1],AL            ; STORE SECOND BYTE
F5FA AC                         LODSB                           ; GET CODE POINT
F5FB E8C500                     CALL    S21
F5FE 23C3                       AND     AX,BX                   ; CONVERT TO COLOR
F600 F6C280                     TEST    DL,80H                  ; AGAIN, IS THIS XOR FUNCTION
F603 740A                       JZ      S11                     ; NO, JUST STORE THE VALUES
F605 2632A50020                 XOR     AH,ES:[DI+2000H]        ; FUNCTION WITH FIRST HALF
F60A 2632850120                 XOR     AL,ES:[DI+2001H]        ; AND WITH SECOND HALF
F60F                    S11:
F60F 2688A50020                 MOV     ES:[DI+2000H],AH
F614 2688850120                 MOV     ES:[DI+2000H+1],AL      ; STORE IN SECOND PORTION OF BUFFER
F619 83C750                     ADD     DI,80                   ; POINT TO NEXT LOCATION
F61C FECE                       DEC     DH
F61E 75C1                       JNZ     S9                      ; KEEP GOING
F620 5E                         POP     SI                      ; RECOVER CODE PONTER
F621 5F                         POP     DI                      ; RECOVER REGEN POINTER
F622 47                         INC     DI                      ; POINT TO NEXT CHAR POSITION
F623 47                         INC     DI
F624 E2B7                       LOOP    S8                      ; MORE TO WRITE
F626 E99CFB                     JMP     VIDEO_RETURN
                        GRAPHICS_WRITE  ENDP
                        ;------------------------
                        ; GRAPHICS READ         :
                        ;------------------------
F629                    GRAPHICS_READ   PROC    NEAR
F629 E8D600                     CALL    S26                     ; CONVERTED TO OFFSET IN REGEN
F62C 8BF0                       MOV     SI,AX                   ; SAVE IN SI
F62E 83EC08                     SUB     SP,8                    ; ALLOCATE SPACE TO SAVE THE
                                                                ;  READ CODE POINT
F631 8BEC                       MOV     BP,SP                   ; POINTER TO SAVE AREA

                        ;----- DETERMINE GRAPHICS MODES

```

```
LOC OBJ              SOURCE

F633 803E490006            CMP     CRT_MODE,6
F638 06                    PUSH    ES
F639 1F                    POP     DS                     ; POINT TO REGEN SEGMENT
F63A 721A                  JC      S13                    ; MEDIUM RESOLUTION

                   ;----- HIGH RESOLUTION READ

                   ;----- GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT

F63C B604                  MOV     DH,4                   ; NUMBER OF PASSES
F63E               S12:
F63E 8A04                  MOV     AL,[SI]                ; GET FIRST BYTE
F640 884600                MOV     [BP],AL                ; SAVE IN STORAGE AREA
F643 45                    INC     BP                     ; NEXT LOCATION
F644 8A840020              MOV     AL,[SI+2000H]          ; GET LOWER REGION BYTE
F648 884600                MOV     [BP],AL                ; ADJUST AND STORE
F64B 45                    INC     BP
F64C 83C650                ADD     SI,80                  ; POINTER INTO REGEN
F64F FECE                  DEC     DH                     ; LOOP CONTROL
F651 75EB                  JNZ     S12                    ; DO IT SOME MORE
F653 EB1790                JMP     S15                    ; GO MATCH THE SAVED CODE POINTS

                   ;----- MEDIUM RESOLUTION READ

F656               S13:                                   ; MED_RES_READ
F656 D1E6                  SAL     SI,1                   ; OFFSET*2 SINCE 2 BYTES/CHAR
F658 B604                  MOV     DH,4                   ; NUMBER OF PASSES
F65A               S14:
F65A E88800                CALL    S23                    ; GET PAIR BYTES FROM REGEN
                                                          ;  INTO SINGLE SAVE
F65D 81C60020              ADD     SI,2000H               ; GO TO LOWER REGION
F661 E88100                CALL    S23                    ; GET THIS PAIR INTO SAVE
F664 81EEB01F              SUB     SI,2000H-80            ; ADJUST POINTER BACK INTO UPPER
F668 FECE                  DEC     DH
F66A 75EE                  JNZ     S14                    ; KEEP GOING UNTIL ALL 8 DONE

                   ;----- SAVE AREA HAS CHARACTER IN IT, MATCH IT

F66C               S15:                                   ; FIND_CHAR
F66C BF6EFA90              MOV     DI,OFFSET CRT_CHAR_GEN ; ESTABLISH ADDRESSING
F670 0E                    PUSH    CS
F671 07                    POP     ES                     ; CODE POINTS IN CS
F672 83ED08                SUB     BP,8                   ; ADJUST POINTER TO BEGINNING
                                                          ;  OF SAVE AREA
F675 8BF5                  MOV     SI,BP
F677 FC                    CLD                            ; ENSURE DIRECTION
F678 B000                  MOV     AL,0                   ; CURRENT CODE POINT BEING MATCHED
F67A               S16:
F67A 16                    PUSH    SS                     ; ESTABLISH ADDRESSING TO STACK
F67B 1F                    POP     DS                     ; FOR THE STRING COMPARE
F67C BA8000                MOV     DX,128                 ; NUMBER TO TEST AGAINST
F67F               S17:
F67F 56                    PUSH    SI                     ; SAVE SAVE AREA POINTER
F680 57                    PUSH    DI                     ; SAVE CODE POINTER
F681 B90800                MOV     CX,8                   ; NUMBER OF BYTES TO MATCH
F684 F3                    REPE    CMPSB                  ; COMPARE THE 8 BYTES
F685 A6
F686 5F                    POP     DI                     ; RECOVER THE POINTERS
F687 5E                    POP     SI
F688 741E                  JZ      S18                    ; IF ZERO FLAG SET, THEN MATCH OCCURRED
F68A FEC0                  INC     AL                     ; NO MATCH, MOVE ON TO NEXT
F68C 83C708                ADD     DI,8                   ; NEXT CODE POINT
F68F 4A                    DEC     DX                     ; LOOP CONTROL
F690 75ED                  JNZ     S17                    ; DO ALL OF THEM

                   ;----- CHAR NOT MATCHED, MIGHT BE IN USER SUPPLIED SECOND HALF

F692 3C00                  CMP     AL,0                   ; AL <> 0 IF ONLY 1ST HALF SCANNED
F694 7412                  JE      S18                    ; IF = 0, THEN ALL HAS BEEN SCANNED
F696 2BC0                  SUB     AX,AX
F698 8ED8                  MOV     DS,AX                  ; ESTABLISH ADDRESSING TO VECTOR
                           ASSUME  DS:ABS0
F69A C43E7C00              LES     DI,EXT_PTR             ; GET POINTER
F69E 8CC0                  MOV     AX,ES                  ; SEE IF THE POINTER REALLY EXISTS
F6A0 0BC7                  OR      AX,DI                  ; IF ALL 0, THEN DOESN'T EXIST
F6A2 7404                  JZ      S18                    ; NO SENSE LOOKING
F6A4 B080                  MOV     AL,128                 ; ORIGIN FOR SECOND HALF
```

Appendix A

```
LOC OBJ         SOURCE

F6A6 EBD2               JMP     S16                     ; GO BACK AND TRY FOR IT
                        ASSUME  DS:DATA

                ;----- CHARACTER IS FOUND ( AL=0 IF NOT FOUND )

F6A8            S18:
F6A8 83C408             ADD     SP,8                    ; READJUST THE STACK, THROW AWAY SAVE
F6AB E917FB             JMP     VIDEO_RETURN            ; ALL DONE
                GRAPHICS_READ   ENDP
                ;--------------------------------------------------------
                ; EXPAND_MED_COLOR                                      :
                ;       THIS ROUTINE EXPANDS THE LOW 2 BITS IN BL TO    :
                ;       FILL THE ENTIRE BX REGISTER                     :
                ; ENTRY                                                 :
                ;       BL = COLOR TO BE USED ( LOW 2 BITS )            :
                ; EXIT                                                  :
                ;       BX = COLOR TO BE USED ( 8 REPLICATIONS OF THE   :
                ;       2 COLOR BITS )                                  :
                ;--------------------------------------------------------
F6AE            S19     PROC    NEAR
F6AE 80E303             AND     BL,3                    ; ISOLATE THE COLOR BITS
F6B1 8AC3               MOV     AL,BL                   ; COPY TO AL
F6B3 51                 PUSH    CX                      ; SAVE REGISTER
F6B4 B90300             MOV     CX,3                    ; NUMBER OF TIMES TO DO THIS
F6B7            S20:
F6B7 D0E0               SAL     AL,1
F6B9 D0E0               SAL     AL,1                    ; LEFT SHIFT BY 2
F6BB 0AD8               OR      BL,AL                   ; ANOTHER COLOR VERSION INTO BL
F6BD E2F8               LOOP    S20                     ; FILL ALL OF BL
F6BF 8AFB               MOV     BH,BL                   ; FILL UPPER PORTION
F6C1 59                 POP     CX                      ; REGISTER BACK
F6C2 C3                 RET                             ; ALL DONE
                S19     ENDP
                ;--------------------------------------------------------
                ; EXPAND_BYTE                                           :
                ;       THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES   :
                ;       ALL OF THE BITS, TURNING THE 8 BITS INTO        :
                ;       16 BITS. THE RESULT IS LEFT IN AX               :
                ;--------------------------------------------------------
F6C3            S21     PROC    NEAR
F6C3 52                 PUSH    DX                      ; SAVE REGISTERS
F6C4 51                 PUSH    CX
F6C5 53                 PUSH    BX
F6C6 2BD2               SUB     DX,DX                   ; RESULT REGISTER
F6C8 B90100             MOV     CX,1                    ; MASK REGISTER
F6CB            S22:
F6CB 8BD8               MOV     BX,AX                   ; BASE INTO TEMP
F6CD 23D9               AND     BX,CX                   ; USE MASK TO EXTRACT A BIT
F6CF 0BD3               OR      DX,BX                   ; PUT INTO RESULT REGISTER
F6D1 D1E0               SHL     AX,1
F6D3 D1E1               SHL     CX,1                    ; SHIFT BASE AND MASK BY 1
F6D5 8BD8               MOV     BX,AX                   ; BASE TO TEMP
F6D7 23D9               AND     BX,CX                   ; EXTRACT THE SAME BIT
F6D9 0BD3               OR      DX,BX                   ; PUT INTO RESULT
F6DB D1E1               SHL     CX,1                    ; SHIFT ONLY MASK NOW,
                                                        ;  MOVING TO NEXT BASE
F6DD 73EC               JNC     S22                     ; USE MASK BIT COMING OUT TO TERMINATE
F6DF 8BC2               MOV     AX,DX                   ; RESULT TO PARM REGISTER
F6E1 5B                 POP     BX
F6E2 59                 POP     CX                      ; RECOVER REGISTERS
F6E3 5A                 POP     DX
F6E4 C3                 RET                             ; ALL DONE
                S21     ENDP
                ;--------------------------------------------------------
                ; MED_READ_BYTE                                         :
                ;       THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN   :
                ;       BUFFER, COMPARE AGAINST THE CURRENT FOREGROUND  :
                ;       COLOR, AND PLACE THE CORRESPONDING ON/OFF BIT   :
                ;       PATTERN INTO THE CURRENT POSITION IN THE SAVE   :
                ;       AREA                                            :
                ; ENTRY                                                 :
                ;       SI,DS = POINTER TO REGEN AREA OF INTEREST       :
                ;       BX = EXPANDED FOREGROUND COLOR                  :
                ;       BP = POINTER TO SAVE AREA                       :
                ; EXIT                                                  :
                ;       BP IS INCREMENT AFTER SAVE                      :
                ;--------------------------------------------------------
```

```
LOC OBJ          LINE    SOURCE

F6E5                     S23     PROC    NEAR
F6E5 8A24                        MOV     AH,[SI]                 ; GET FIRST BYTE
F6E7 8A4401                      MOV     AL,[SI+1]               ; GET SECOND BYTE
F6EA B900C0                      MOV     CX,0C000H               ; 2 BIT MASK TO TEST THE ENTRIES
F6ED B200                        MOV     DL,0                    ; RESULT REGISTER
F6EF                     S24:
F6EF 85C1                        TEST    AX,CX                   ; IS THIS SECTION BACKGROUND?
F6F1 F8                          CLC                             ; CLEAR CARRY IN HOPES THAT IT IS
F6F2 7401                        JZ      S25                     ; IF ZERO, IT IS BACKGROUND
F6F4 F9                          STC                             ; WASN'T, SO SET CARRY
F6F5 D0D2                S25:    RCL     DL,1                    ; MOVE THAT BIT INTO THE RESULT
F6F7 D1E9                        SHR     CX,1
F6F9 D1E9                        SHR     CX,1                    ; MOVE THE MASK TO THE RIGHT BY 2 BITS
F6FB 73F2                        JNC     S24                     ; DO IT AGAIN IF MASK DIDN'T FALL OUT
F6FD 885600                      MOV     [BP],DL                 ; STORE RESULT IN SAVE AREA
F700 45                          INC     BP                      ; ADJUST POINTER
F701 C3                          RET                             ; ALL DONE
                         S23     ENDP
                         ;------------------------------------------------
                         ; V4_POSITION                                   :
                         ;       THIS ROUTINE TAKES THE CURSOR POSITION  :
                         ;       CONTAINED IN THE MEMORY LOCATION, AND   :
                         ;       CONVERTS IT INTO AN OFFSET INTO THE     :
                         ;       REGEN BUFFER, ASSUMING ONE BYTE/CHAR.   :
                         ;       FOR MEDIUM RESOLUTION GRAPHICS,         :
                         ;       THE NUMBER MUST BE DOUBLED.             :
                         ; ENTRY                                         :
                         ;       NO REGISTERS, MEMORY LOCATION           :
                         ;       CURSOR_POSN IS USED                     :
                         ; EXIT                                          :
                         ;       AX CONTAINS OFFSET INTO REGEN BUFFER    :
                         ;------------------------------------------------
F702                     S26     PROC    NEAR
F702 A15000                      MOV     AX,CURSOR_POSN          ; GET CURRENT CURSOR
F705                     GRAPH_POSN      LABEL   NEAR
F705 53                          PUSH    BX                      ; SAVE REGISTER
F706 8BD8                        MOV     BX,AX                   ; SAVE A COPY OF CURRENT CURSOR
F708 8AC4                        MOV     AL,AH                   ; GET ROWS TO AL
F70A F6264A00                    MUL     BYTE PTR CRT_COLS       ; MULTIPLY BY BYTES/COLUMN
F70E D1E0                        SHL     AX,1                    ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
F710 D1E0                        SHL     AX,1
F712 2AFF                        SUB     BH,BH                   ; ISOLATE COLUMN VALUE
F714 03C3                        ADD     AX,BX                   ; DETERMINE OFFSET
F716 5B                          POP     BX                      ; RECOVER POINTER
F717 C3                          RET                             ; ALL DONE
                         S26     ENDP
                         ;----------------------------------------------------------------------
                         ; WRITE_TTY                                                           :
                         ;       THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE VIDEO :
                         ;       CARD. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT CURSOR     :
                         ;       POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION. IF THE :
                         ;       CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN IS SET  :
                         ;       TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW VALUE    :
                         ;       LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW, FIRST  :
                         ;       COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE. WHEN    :
                         ;       THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE NEWLY :
                         ;       BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS  :
                         ;       LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE,   :
                         ;       THE 0 COLOR IS USED.                                           :
                         ; ENTRY                                                               :
                         ;       (AH) = CURRENT CRT MODE                                        :
                         ;       (AL) = CHARACTER TO BE WRITTEN                                 :
                         ;        NOTE THAT BACK SPACE, CAR RET, BELL AND LINE FEED ARE HANDLED :
                         ;        AS COMMANDS RATHER THAN AS DISPLAYABLE GRAPHICS               :
                         ;       (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A       :
                         ;        GRAPHICS MODE                                                 :
                         ; EXIT                                                                :
                         ;       ALL REGISTERS SAVED                                            :
                         ;----------------------------------------------------------------------
                                 ASSUME  CS:CODE,DS:DATA
F718                     WRITE_TTY       PROC    NEAR
F718 50                          PUSH    AX                      ; SAVE REGISTERS
F719 50                          PUSH    AX                      ; SAVE CHAR TO WRITE
F71A B403                        MOV     AH,3
F71C 8A3E6200                    MOV     BH,ACTIVE_PAGE          ; GET THE CURRENT ACTIVE PAGE
F720 CD10                        INT     10H                     ; READ THE CURRENT CURSOR POSITION
F722 58                          POP     AX                      ; RECOVER CHAR
```

Appendix A

```
LOC OBJ          SOURCE

                 ;----- DX NOW HAS THE CURRENT CURSOR POSITION

F723 3C08                CMP     AL,8                    ; IS IT A BACKSPACE
F725 7452                JE      U8                      ; BACK_SPACE
F727 3C0D                CMP     AL,0DH                  ; IS IT CARRIAGE RETURN
F729 7457                JE      U9                      ; CAR_RET
F72B 3C0A                CMP     AL,0AH                  ; IS IT A LINE FEED
F72D 7457                JE      U10                     ; LINE_FEED
F72F 3C07                CMP     AL,07H                  ; IS IT A BELL
F731 745A                JE      U11                     ; BELL

                 ;----- WRITE THE CHAR TO THE SCREEN


F733 B40A                MOV     AH,10                   ; WRITE CHAR ONLY
F735 B90100              MOV     CX,1                    ; ONLY ONE CHAR
F738 CD10                INT     10H                     ; WRITE THE CHAR

                 ;----- POSITION THE CURSOR FOR NEXT CHAR

F73A FEC2                INC     DL
F73C 3A164A00            CMP     DL,BYTE PTR CRT_COLS    ; TEST FOR COLUMN OVERFLOW
F740 7533                JNZ     U7                      ; SET_CURSOR
F742 B200                MOV     DL,0                    ; COLUMN FOR CURSOR
F744 80FE18              CMP     DH,24
F747 752A                JNZ     U6                      ; SET_CURSOR_INC

                 ;----- SCROLL REQUIRED

F749             U1:
F749 B402                MOV     AH,2
F74B CD10                INT     10H                     ; SET THE CURSOR

                 ;----- DETERMINE VALUE TO FILL WITH DURING SCROLL

F74D A04900              MOV     AL,CRT_MODE             ; GET THE CURRENT MODE
F750 3C04                CMP     AL,4
F752 7206                JC      U2                      ; READ-CURSOR
F754 3C07                CMP     AL,7
F756 B700                MOV     BH,0                    ; FILL WITH BACKGROUND
F758 7506                JNE     U3                      ; SCROLL-UP
F75A             U2:                                     ; READ-CURSOR
F75A B408                MOV     AH,8
F75C CD10                INT     10H                     ; READ CHAR/ATTR AT CURRENT CURSOR
F75E 8AFC                MOV     BH,AH                   ; STORE IN BH
F760             U3:                                     ; SCROLL-UP
F760 B80106              MOV     AX,601H                 ; SCROLL ONE LINE
F763 2BC9                SUB     CX,CX                   ; UPPER LEFT CORNER
F765 B618                MOV     DH,24                   ; LOWER RIGHT ROW
F767 8A164A00            MOV     DL,BYTE PTR CRT_COLS    ; LOWER RIGHT COLUMN
F76B FECA                DEC     DL
F76D             U4:                                     ; VIDEO-CALL-RETURN
F76D CD10                INT     10H                     ; SCROLL UP THE SCREEN
F76F             U5:                                     ; TTY-RETURN
F76F 58                  POP     AX                      ; RESTORE THE CHARACTER
F770 E952FA              JMP     VIDEO_RETURN            ; RETURN TO CALLER
F773             U6:                                     ; SET-CURSOR-INC
F773 FEC6                INC     DH                      ; NEXT ROW
F775             U7:                                     ; SET-CURSOR
F775 B402                MOV     AH,2
F777 EBF4                JMP     U4                      ; ESTABLISH THE NEW CURSOR

                 ;----- BACK SPACE FOUND

F779             U8:
F779 80FA00              CMP     DL,0                    ; ALREADY AT END OF LINE
F77C 74F7                JE      U7                      ; SET_CURSOR
F77E FECA                DEC     DL                      ; NO -- JUST MOVE IT BACK
F780 EBF3                JMP     U7                      ; SET_CURSOR

                 ;----- CARRIAGE RETURN FOUND

F782             U9:
F782 B200                MOV     DL,0                    ; MOVE TO FIRST COLUMN
F784 EBEF                JMP     U7                      ; SET_CURSOR

```

```
LOC OBJ              SOURCE

                     ;----- LINE FEED FOUND

F786                 U10:
F786 80FE18                  CMP     DH,24                   ; BOTTOM OF SCREEN
F789 75E8                    JNE     U6                      ; YES, SCROLL THE SCREEN
F78B EBBC                    JMP     U1                      ; NO, JUST SET THE CURSOR

                     ;----- BELL FOUND

F78D                 U11:
F78D B302                    MOV     BL,2                    ; SET UP COUNT FOR BEEP
F78F E871EE                  CALL    BEEP                    ; SOUND THE POD BELL
F792 EBD8                    JMP     U5                      ; TTY_RETURN
                     WRITE_TTY       ENDP
                     ;--------------------------------------------------------------
                     ; LIGHT PEN                                                    :
                     ;       THIS ROUTINE TESTS THE LIGHT PEN SWITCH AND THE LIGHT  :
                     ;       PEN TRIGGER. IF BOTH ARE SET, THE LOCATION OF THE LIGHT :
                     ;       PEN IS DETERMINED. OTHERWISE, A RETURN WITH NO         :
                     ;       INFORMATION IS MADE.                                   :
                     ; ON EXIT                                                      :
                     ;       (AH) = 0 IF NO LIGHT PEN INFORMATION IS AVAILABLE      :
                     ;               BX,CX,DX ARE DESTROYED                         :
                     ;       (AH) = 1 IF LIGHT PEN IS AVAILABLE                     :
                     ;               (DH,DL) = ROW,COLUMN OF CURRENT LIGHT PEN      :
                     ;                       POSITION                               :
                     ;               (CH) = RASTER POSITION                         :
                     ;               (BX) = BEST GUESS AT PIXEL HORIZONTAL POSITION :
                     ;--------------------------------------------------------------
                             ASSUME  CS:CODE,DS:DATA
                     ;----- SUBTRACT_TABLE
F794                 V1      LABEL   BYTE
F794 03                      DB      3,3,5,5,3,3,3,4 ;
F795 03
F796 05
F797 05
F798 03
F799 03
F79A 03
F79B 04
F79C                 READ_LPEN       PROC    NEAR

                     ;----- WAIT FOR LIGHT PEN TO BE DEPRESSED

F79C B400                    MOV     AH,0                    ; SET NO LIGHT PEN RETURN CODE
F79E 8B166300                MOV     DX,ADDR_6845            ; GET BASE ADDRESS OF 6845
F7A2 83C206                  ADD     DX,6                    ; POINT TO STATUS REGISTER
F7A5 EC                      IN      AL,DX                   ; GET STATUS REGISTER
F7A6 A804                    TEST    AL,4                    ; TEST LIGHT PEN SWITCH
F7A8 757E                    JNZ     V6                      ; NOT SET, RETURN

                     ;----- NOW TEST FOR LIGHT PEN TRIGGER

F7AA A802                    TEST    AL,2                    ; TEST LIGHT PEN TRIGGER
F7AC 7503                    JNZ     V7A                     ; RETURN WITHOUT RESETTING TRIGGER
F7AE E98100                  JMP     V7

                     ;----- TRIGGER HAS BEEN SET, READ THE VALUE IN

F7B1                 V7A:
F7B1 B410                    MOV     AH,16                   ; LIGHT PEN REGISTERS ON 6845

                     ;----- INPUT REGS POINTED TO BY AH, AND CONVERT TO ROW COLUMN IN DX

F7B3 8B166300                MOV     DX,ADDR_6845            ; ADDRESS REGISTER FOR 6845
F7B7 8AC4                    MOV     AL,AH                   ; REGISTER TO READ
F7B9 EE                      OUT     DX,AL                   ; SET IT UP
F7BA 42                      INC     DX                      ; DATA REGISTER
F7BB EC                      IN      AL,DX                   ; GET THE VALUE
F7BC 8AE8                    MOV     CH,AL                   ; SAVE IN CX
F7BE 4A                      DEC     DX                      ; ADDRESS REGISTER
F7BF FEC4                    INC     AH
F7C1 8AC4                    MOV     AL,AH                   ; SECOND DATA REGISTER
F7C3 EE                      OUT     DX,AL
F7C4 42                      INC     DX                      ; POINT TO DATA REGISTER
F7C5 EC                      IN      AL,DX                   ; GET SECOND DATA VALUE
F7C6 8AE5                    MOV     AH,CH                   ; AX HAS INPUT VALUE
```

```
LOC OBJ                SOURCE

                       ;----- AX HAS THE VALUE READ IN FROM THE 6845

F7C8 8A1E4900                MOV     BL,CRT_MODE
F7CC 2AFF                    SUB     BH,BH                  ; MODE VALUE TO BX
F7CE 2E8A9F94F7              MOV     BL,CS:V1[BX]           ; DETERMINE AMOUNT TO SUBTRACT
F7D3 2BC3                    SUB     AX,BX                  ; TAKE IT AWAY
F7D5 8B1E4E00                MOV     BX,CRT_START
F7D9 D1EB                    SHR     BX,1
F7DB 2BC3                    SUB     AX,BX
F7DD 7902                    JNS     V2                     ; IF POSITIVE, DETERMINE MODE
F7DF 2BC0                    SUB     AX,AX                  ; <0 PLAYS AS 0

                       ;----- DETERMINE MODE OF OPERATION

F7E1                   V2:                                  ; DETERMINE_MODE
F7E1 B103                    MOV     CL,3                   ; SET *8 SHIFT COUNT
F7E3 803E490004              CMP     CRT_MODE,4             ; DETERMINE IF GRAPHICS OR ALPHA
F7E8 722A                    JB      V4                     ; ALPHA_PEN
F7EA 803E490007              CMP     CRT_MODE,7
F7EF 7423                    JE      V4                     ; ALPHA_PEN

                       ;----- GRAPHICS MODE

F7F1 B228                    MOV     DL,40                  ; DIVISOR FOR GRAPHICS
F7F3 F6F2                    DIV     DL                     ; DETERMINE ROW(AL) AND COLUMN(AH)
                                                            ;  AL RANGE 0-99, AH RANGE 0-39

                       ;----- DETERMINE GRAPHIC ROW POSITION

F7F5 8AE8                    MOV     CH,AL                  ; SAVE ROW VALUE IN CH
F7F7 02ED                    ADD     CH,CH                  ; *2 FOR EVEN/ODD FIELD
F7F9 8ADC                    MOV     BL,AH                  ; COLUMN VALUE TO BX
F7FB 2AFF                    SUB     BH,BH                  ; MULTIPLY BY 8 FOR MEDIUM RES
F7FD 803E490006              CMP     CRT_MODE,6             ; DETERMINE MEDIUM OR HIGH RES
F802 7504                    JNE     V3                     ; NOT_HIGH_RES
F804 B104                    MOV     CL,4                   ; SHIFT VALUE FOR HIGH RES
F806 D0E4                    SAL     AH,1                   ; COLUMN VALUE TIMES 2 FOR HIGH RES
F808                   V3:                                  ; NOT_HIGH_RES
F808 D3E3                    SHL     BX,CL                  ; MULTIPLY *16 FOR HIGH RES

                       ;----- DETERMINE ALPHA CHAR POSITION

F80A 8AD4                    MOV     DL,AH                  ; COLUMN VALUE FOR RETURN
F80C 8AF0                    MOV     DH,AL                  ; ROW VALUE
F80E D0EE                    SHR     DH,1                   ; DIVIDE BY 4
F810 D0EE                    SHR     DH,1                   ;  FOR VALUE IN 0-24 RANGE
F812 EB12                    JMP     SHORT V5               ; LIGHT_PEN_RETURN_SET

                       ;----- ALPHA MODE ON LIGHT PEN

F814                   V4:                                  ; ALPHA_PEN
F814 F6364A00                DIV     BYTE PTR CRT_COLS      ; DETERMINE ROW,COLUMN VALUE
F818 8AF0                    MOV     DH,AL                  ; ROWS TO DH
F81A 8AD4                    MOV     DL,AH                  ; COLS TO DL
F81C D2E0                    SAL     AL,CL                  ; MULTIPLY ROWS * 8
F81E 8AE8                    MOV     CH,AL                  ; GET RASTER VALUE TO RETURN REG
F820 8ADC                    MOV     BL,AH                  ; COLUMN VALUE
F822 32FF                    XOR     BH,BH                  ;  TO BX
F824 D3E3                    SAL     BX,CL
F826                   V5:                                  ; LIGHT_PEN_RETURN_SET
F826 B401                    MOV     AH,1                   ; INDICATE EVERTHING SET
F828                   V6:                                  ; LIGHT_PEN_RETURN
F828 52                      PUSH    DX                     ; SAVE RETURN VALUE (IN CASE)
F829 8B166300                MOV     DX,ADDR_6845           ; GET BASE ADDRESS
F82D 83C207                  ADD     DX,7                   ; POINT TO RESET PARM
F830 EE                      OUT     DX,AL                  ; ADDRESS, NOT DATA, IS IMPORTANT
F831 5A                      POP     DX                     ; RECOVER VALUE
F832                   V7:                                  ; RETURN_NO_RESET
F832 5F                      POP     DI
F833 5E                      POP     SI
F834 1F                      POP     DS                     ; DISCARD SAVED BX,CX,DX
F835 1F                      POP     DS
F836 1F                      POP     DS

F837 1F                      POP     DS
F838 07                      POP     ES
```

```
LOC OBJ           LINE    SOURCE

F839 CF                          IRET
                          READ_LPEN       ENDP

                          ;--- INT 12 -------------------------------------------------------------
                          ; MEMORY_SIZE_DET                                                         :
                          ;       THIS ROUTINE DETERMINES THE AMOUNT OF MEMORY IN THE SYSTEM        :
                          ;       AS REPRESENTED BY THE SWITCHES ON THE PLANAR.  NOTE THAT THE      :
                          ;       SYSTEM MAY NOT BE ABLE TO USE I/O MEMORY UNLESS THERE IS A FULL  :
                          ;       COMPLEMENT OF 64K BYTES ON THE PLANAR.                            :
                          ; INPUT                                                                   :
                          ;       NO REGISTERS                                                      :
                          ;       THE MEMORY_SIZE VARIABLE IS SET DURING POWER ON DIAGNOSTICS       :
                          ;        ACCORDING TO THE FOLLOWING HARDWARE ASSUMPTIONS:                 :
                          ;       PORT 60 BITS 3,2 = 00 - 16K BASE RAM                              :
                          ;                          01 - 32K BASE RAM                              :
                          ;                          10 - 48K BASE RAM                              :
                          ;                          11 - 64K BASE RAM                              :
                          ;       PORT 62 BITS 3-0 INDICATE AMOUNT OF I/O RAM IN 32K INCREMENTS    :
                          ;               E.G., 0000 - NO RAM IN I/O CHANNEL                        :
                          ;                     0010 - 64K RAM IN I/O CHANNEL, ETC.                 :
                          ; OUTPUT                                                                  :
                          ;       (AX) = NUMBER OF CONTIGUOUS 1K BLOCKS OF MEMORY                   :
                          ;-------------------------------------------------------------------------
                                  ASSUME  CS:CODE,DS:DATA
F841                              ORG     0F841H
F841                      MEMORY_SIZE_DET PROC    FAR
F841 FB                           STI                             ; INTERRUPTS BACK ON
F842 1E                           PUSH    DS                      ; SAVE SEGMENT
F843 E8F806                       CALL    DDS
F846 A11300                       MOV     AX,MEMORY_SIZE          ; GET VALUE
F849 1F                           POP     DS                      ; RECOVER SEGMENT
F84A CF                           IRET                            ; RETURN TO CALLER
                          MEMORY_SIZE_DET  ENDP

                          ;--- INT 11 ---------------------------------------------------
                          ; EQUIPMENT DETERMINATION                                       :
                          ;       THIS ROUTINE ATTEMPTS TO DETERMINE WHAT OPTIONAL        :
                          ;       DEVICES ARE ATTACHED TO THE SYSTEM.                     :
                          ; INPUT                                                         :
                          ;       NO REGISTERS                                            :
                          ;       THE EQUIP_FLAG VARIABLE IS SET DURING THE POWER ON      :
                          ;       DIAGNOSTICS USING THE FOLLOWING HARDWARE ASSUMPTIONS:   :
                          ;       PORT 60  = LOW ORDER BYTE OF EQUPMENT                   :
                          ;       PORT 3FA = INTERRUPT ID REGISTER OF 8250                :
                          ;               BITS 7-3 ARE ALWAYS 0                           :
                          ;       PORT 378 = OUTPUT PORT OF PRINTER -- 8255 PORT THAT     :
                          ;               CAN BE READ AS WELL AS WRITTEN                  :
                          ; OUTPUT                                                        :
                          ;       (AX) IS SET, BIT SIGNIFICANT, TO INDICATE ATTACHED I/O  :
                          ;       BIT 15,14 = NUMBER OF PRINTERS ATTACHED                 :
                          ;       BIT 13 NOT USED                                         :
                          ;       BIT 12 = GAME I/O ATTACHED                              :
                          ;       BIT 11,10,9 = NUMBER OF RS232 CARDS ATTACHED            :
                          ;       BIT 8 UNUSED                                            :
                          ;       BIT 7,6 = NUMBER OF DISKETTE DRIVES                     :
                          ;               00=1, 01=2, 10=3, 11=4 ONLY IF BIT 0 = 1        :
                          ;       BIT 5,4 = INITIAL VIDEO MODE                            :
                          ;                       00 - UNUSED                             :
                          ;                       01 - 40X25 BW USING COLOR CARD          :
                          ;                       10 - 80X25 BW USING COLOR CARD          :
                          ;                       11 - 80X25 BW USING BW CARD             :
                          ;       BIT 3,2 = PLANAR RAM SIZE (00=16K,01=32K,10=48K,11=64K) :
                          ;       BIT 1 NOT USED                                          :
                          ;       BIT 0 = IPL FROM DISKETTE -- THIS BIT INDICATES THAT    :
                          ;               THERE ARE DISKETTE DRIVES ON THE SYSTEM         :
                          ;                                                               :
                          ;       NO OTHER REGISTERS AFFECTED                             :
                          ;---------------------------------------------------------------
                                  ASSUME  CS:CODE,DS:DATA
F84D                              ORG     0F84DH
F84D                      EQUIPMENT       PROC    FAR
F84D FB                           STI                             ; INTERRUPTS BACK ON
F84E 1E                           PUSH    DS                      ; SAVE SEGMENT REGISTER
F84F E8EC06                       CALL    DDS
F852 A11000                       MOV     AX,EQUIP_FLAG           ; GET THE CURRENT SETTINGS
F855 1F                           POP     DS                      ; RECOVER SEGMENT
F856 CF                           IRET                            ; RETURN TO CALLER
```

Appendix A

```
                EQUIPMENT       ENDP

                ;--- INT 15 -----------------------------------------------------
                ; CASSETTE I/O                                                  :
                ;       (AH) = 0  TURN CASSETTE MOTOR ON                         :
                ;       (AH) = 1  TURN CASSETTE MOTOR OFF                        :
                ;       (AH) = 2  READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE   :
                ;               (ES,BX) = POINTER TO DATA BUFFER                 :
                ;               (CX) = COUNT OF BYTES TO READ                    :
                ; ON EXIT                                                        :
                ;       (ES,BX) = POINTER TO LAST BYTE READ + 1                  :
                ;       (DX) = COUNT OF BYTES ACTUALLY READ                      :
                ;       (CY) = 0 IF NO ERROR OCCURRED                            :
                ;            = 1 IF ERROR OCCURRED                               :
                ;       (AH) = ERROR RETURN IF (CY)= 1                           :
                ;               = 01 IF CRC ERROR WAS DETECTED                   :
                ;               = 02 IF DATA TRANSITIONS ARE LOST                :
                ;               = 04 IF NO DATA WAS FOUND                        :
                ;       (AH) = 3 WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE     :
                ;               (ES,BX) = POINTER TO DATA BUFFER                 :
                ;               (CX) = COUNT OF BYTES TO WRITE                   :
                ; ON EXIT                                                        :
                ;       (EX,BX) = POINTER TO LAST BYTE WRITTEN + 1               :
                ;       (CX) = 0                                                 :
                ;       (AH) = ANY OTHER THAN ABOVE VALUES CAUSES (CY)= 1        :
                ;               AND (AH)= 80 TO BE RETURNED (INVALID COMMAND).   :
                ;----------------------------------------------------------------
                        ASSUME  DS:DATA,ES:NOTHING,SS:NOTHING,CS:CODE
F859                    ORG     0F859H
F859            CASSETTE_IO     PROC    FAR
F859 FB                 STI                             ; INTERRUPTS BACK ON
F85A 1E                 PUSH    DS                      ; ESTABLISH ADDRESSING TO DATA
F85B E8E006             CALL    DDS
F85E 802671007F         AND     BIOS_BREAK, 7FH         ; MAKE SURE BREAK FLAG IS OFF
F863 E80400             CALL    W1                      ; CASSETTE_IO_CONT
F866 1F                 POP     DS
F867 CA0200             RET     2                       ; INTERRUPT RETURN
                CASSETTE_IO     ENDP
F86A            W1      PROC    NEAR
                ;--------------------------------------------------------
                ; PURPOSE:                                              :
                ;  TO CALL APPROPRIATE ROUTINE DEPENDING ON REG AH      :
                ;                                                       :
                ;  AH           ROUTINE                                 :
                ;--------------------------------------------------------
                ;  0            MOTOR ON                                :
                ;  1            MOTOR OFF                               :
                ;  2            READ CASSETTE BLOCK                     :
                ;  3            WRITE CASSETTE BLOCK                    :
                ;--------------------------------------------------------
F86A 0AE4               OR      AH,AH                   ; TURN ON MOTOR?
F86C 7413               JZ      MOTOR_ON                ; YES, DO IT
F86E FECC               DEC     AH                      ; TURN OFF MOTOR?
F870 7418               JZ      MOTOR_OFF               ; YES, DO IT
F872 FECC               DEC     AH                      ; READ CASSETTE BLOCK?
F874 741A               JZ      READ_BLOCK              ; YES, DO IT
F876 FECC               DEC     AH                      ; WRITE CASSETTE BLOCK?
F878 7503               JNZ     W2                      ;  NOT_DEFINED
F87A E92401             JMP     WRITE_BLOCK             ; YES, DO IT
F87D            W2:                                     ; COMMAND NOT DEFINED
F87D B480               MOV     AH,080H                 ; ERROR, UNDEFINED OPERATION
F87F F9                 STC                             ; ERROR FLAG
F880 C3                 RET
                W1      ENDP
F881            MOTOR_ON        PROC    NEAR
                ;----------------------------------------
                ; PURPOSE:                              :
                ;       TO TURN ON CASSETTE MOTOR       :
                ;----------------------------------------
F881 E461               IN      AL,PORT_B               ; READ CASSETTE OUTPUT
F883 24F7               AND     AL,NOT 08H              ; CLEAR BIT TO TURN ON MOTOR
F885            W3:
F885 E661               OUT     PORT_B,AL               ; WRITE IT OUT
F887 2AE4               SUB     AH,AH                   ; CLEAR AH
F889 C3                 RET
                MOTOR_ON        ENDP
F88A            MOTOR_OFF       PROC    NEAR
```

```
LOC OBJ                 SOURCE

                        ;----------------------------------------
                        ; PURPOSE:                               :
                        ;       TO TURN CASSETTE MOTOR OFF       :
                        ;----------------------------------------
F88A E461                       IN      AL,PORT_B               ; READ CASSETTE OUTPUT
F88C 0C08                       OR      AL,08H                  ; SET BIT TO TURN OFF
F88E EBF5                       JMP     W3                      ; WRITE IT, CLEAR ERROR, RETURN
                        MOTOR_OFF       ENDP
F890                    READ_BLOCK      PROC    NEAR
                        ;------------------------------------------------------------
                        ; PURPOSE:                                                   :
                        ;       TO READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE      :
                        ;                                                            :
                        ; ON ENTRY:                                                  :
                        ;       ES IS SEGMENT FOR MEMORY BUFFER (FOR COMPACT CODE)   :
                        ;       BX POINTS TO START OF MEMORY BUFFER                  :
                        ;       CX CONTAINS NUMBER OF BYTES TO READ                  :
                        ; ON EXIT:                                                   :
                        ;       BX POINTS 1 BYTE PAST LAST BYTE PUT IN MEM           :
                        ;       CX CONTAINS DECREMENTED BYTE COUNT                   :
                        ;       DX CONTAINS NUMBER OF BYTES ACTUALLY READ            :
                        ;                                                            :
                        ;       CARRY FLAG IS CLEAR IF NO ERROR DETECTED             :
                        ;       CARRY FLAG IS SET IF CRC ERROR DETECTED              :
                        ;------------------------------------------------------------
F890 53                         PUSH    BX                      ; SAVE BX
F891 51                         PUSH    CX                      ; SAVE CX
F892 56                         PUSH    SI                      ; SAVE SI
F893 BE0700                     MOV     SI, 7                   ; SET UP RETRY COUNT FOR LEADER
F896 E8BF01                     CALL    BEGIN_OP                ; BEGIN BY STARTING MOTOR
F899                    W4:                                     ; SEARCH FOR LEADER
F899 E462                       IN      AL,PORT_C               ; GET INTIAL VALUE
F89B 2410                       AND     AL,010H                 ; MASK OFF EXTRANEOUS BITS
F89D A26B00                     MOV     LAST_VAL,AL             ; SAVE IN LOC LAST_VAL
F8A0 BA7A3F                     MOV     DX,16250                ; # OF TRANSITIONS TO LOOK FOR
F8A3                    W5:                                     ; WAIT_FOR_EDGE
F8A3 F606710080                 TEST    BIOS_BREAK, 80H         ; CHECK FOR BREAK KEY
F8A8 7503                       JNZ     W6A                     ; JUMP IF NO BREAK KEY
                                                                ; JUMP IF BREAK KEY HIT
F8AA                    W6:
F8AA 4A                         DEC     DX
F8AB 7503                       JNZ     W7                      ; JUMP IF BEGINNING OF LEADER
F8AD                    W6A:
F8AD E98400                     JMP     W17                     ; JUMP IF NO LEADER FOUND
F8B0                    W7:
F8B0 E8C600                     CALL    READ_HALF_BIT           ; IGNORE FIRST EDGE
F8B3 E3EE                       JCXZ    W5                      ; JUMP IF NO EDGE DETECTED
F8B5 BA7803                     MOV     DX,0378H                ; CHECK FOR HALF BITS
F8B8 B90002                     MOV     CX,200H                 ; MUST HAVE AT LEAST THIS MANY ONE SIZE
                                                                ; PULSES BEFORE CHCKNG FOR SYNC BIT (0)
F8BB E421                       IN      AL, 021H                ; INTERRUPT MASK REGISTER
F8BD 0C01                       OR      AL,1                    ; DISABLE TIMER INTERRUPTS
F8BF E621                       OUT     021H, AL
F8C1                    W8:                                     ; SEARCH-LDR
F8C1 F606710080                 TEST    BIOS_BREAK, 80H         ; CHECK FOR BREAK KEY
F8C6 756C                       JNZ     W17                     ; JUMP IF BREAK KEY HIT
F8C8 51                         PUSH    CX                      ; SAVE REG CX
F8C9 E8A000                     CALL    READ_HALF_BIT           ; GET PULSE WIDTH
F8CC 0BC9                       OR      CX, CX                  ; CHECK FOR TRANSITION
F8CE 59                         POP     CX                      ; RESTORE ONE BIT COUNTER
F8CF 74C8                       JZ      W4                      ; JUMP IF NO TRANSITION
F8D1 3BD3                       CMP     DX,BX                   ; CHECK PULSE WIDTH
F8D3 E304                       JCXZ    W9                      ; IF CX=0 THEN WE CAN LOOK
                                                                ; FOR SYNC BIT (0)
F8D5 73C2                       JNC     W4                      ; JUMP IF ZERO BIT (NOT GOOD LEADER)
F8D7 E2E8                       LOOP    W8                      ; DEC CX AND READ ANOTHER HALF ONE BIT
F8D9                    W9:                                     ; FIND-SYNC
F8D9 72E6                       JC      W8                      ; JUMP IF ONE BIT (STILL LEADER)

                        ;----- A SYNCH BIT HAS BEEN FOUND.  READ SYN CHARACTER:

F8DB E89B00                     CALL    READ_HALF_BIT           ; SKIP OTHER HALF OF SYNC BIT (0)
F8DE E86A00                     CALL    READ_BYTE               ; READ SYN BYTE
F8E1 3C16                       CMP     AL, 16H                 ; SYNCHRONIZATION CHARACTER
F8E3 7549                       JNE     W16                     ; JUMP IF BAD LEADER FOUND.

                        ;----- GOOD CRC SO READ DATA BLOCK(S)
```

Appendix A

```
LOC OBJ                SOURCE

F8E5 5E                        POP     SI                      ; RESTORE REGS
F8E6 59                        POP     CX
F8E7 5B                        POP     BX
                        ;-----------------------------------------------------------------
                        ; READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE                   :
                        ;                                                                :
                        ; ON ENTRY:                                                      :
                        ;       ES IS SEGMENT FOR MEMORY BUFFER (FOR COMPACT CODE)       :
                        ;       BX POINTS TO START OF MEMORY BUFFER                      :
                        ;       CX CONTAINS NUMBER OF BYTES TO READ                      :
                        ; ON EXIT:                                                       :
                        ;       BX POINTS 1 BYTE PAST LAST BYTE PUT IN MEM               :
                        ;       CX CONTAINS DECREMENTED BYTE COUNT                       :
                        ;       DX CONTAINS NUMBER OF BYTES ACTUALLY READ                :
                        ;-----------------------------------------------------------------
F8E8 51                        PUSH    CX                      ; SAVE BYTE COUNT
F8E9                    W10:                                   ; COME HERE BEFORE EACH
                                                               ; 256 BYTE BLOCK IS READ
F8E9 C7066900FFFF              MOV     CRC_REG,0FFFFH          ; INIT CRC REG
F8EF BA0001                    MOV     DX,256                  ; SET DX TO DATA BLOCK SIZE
F8F2                    W11:                                   ; RD_BLK
F8F2 F606710080                TEST    BIOS_BREAK, 80H         ; CHECK FOR BREAK KEY
F8F7 7523                      JNZ     W13                     ; JUMP IF BREAK KEY HIT
F8F9 E84F00                    CALL    READ_BYTE               ; READ BYTE FROM CASSETTE
F8FC 721E                      JC      W13                     ; CY SET INDICATES NO DATA TRANSITIONS
F8FE E305                      JCXZ    W12                     ; IF WE'VE ALREADY REACHED
                                                               ; END OF MEMORY BUFFER
                                                               ; SKIP REST OF BLOCK
F900 268807                    MOV     ES:[BX],AL              ; STORE DATA BYTE AT BYTE PTR
F903 43                        INC     BX                      ; INC BUFFER PTR
F904 49                        DEC     CX                      ; DEC BYTE COUNTER
F905                    W12:                                   ; LOOP UNTIL DATA BLOCK HAS BEEN
                                                               ;  READ FROM CASSETTE.
F905 4A                        DEC     DX                      ; DEC BLOCK CNT
F906 7FEA                      JG      W11                     ; RD_BLK
F908 E84000                    CALL    READ_BYTE               ; NOW READ TWO CRC BYTES
F90B E83D00                    CALL    READ_BYTE
F90E 2AE4                      SUB     AH,AH                   ; CLEAR AH
F910 813E69000F1D              CMP     CRC_REG,1D0FH           ; IS THE CRC CORRECT
F916 7506                      JNE     W14                     ; IF NOT EQUAL CRC IS BAD
F918 E306                      JCXZ    W15                     ; IF BYTE COUNT IS ZERO
                                                               ; THEN WE HAVE READ ENOUGH
                                                               ; SO WE WILL EXIT
F91A EBCD                      JMP     W10                     ; STILL MORE, SO READ ANOTHER BLOCK
F91C                    W13:                                   ; MISSING-DATA
                                                               ; NO DATA TRANSITIONS SO
F91C B401                      MOV     AH,01H                  ; SET AH=02 TO INDICATE
                                                               ; DATA TIMEOUT
F91E                    W14:                                   ; BAD-CRC
F91E FEC4                      INC     AH                      ; EXIT EARLY ON ERROR
                                                               ; SET AH=01 TO INDICATE CRC ERROR
F920                    W15:                                   ; RD-BLK-EX
F920 5A                        POP     DX                      ; CALCULATE COUNT OF
F921 2BD1                      SUB     DX,CX                   ; DATA BYTES ACTUALLY READ
                                                               ; RETURN COUNT IN REG DX
F923 50                        PUSH    AX                      ; SAVE AX (RET CODE)
F924 F6C490                    TEST    AH, 90H                 ; CHECK FOR ERRORS
F927 7513                      JNZ     W18                     ; JUMP IF ERROR DETECTED
F929 E81F00                    CALL    READ_BYTE               ; READ TRAILER
F92C EB0E                      JMP     SHORT W18               ; SKIP TO TURN OFF MOTOR
F92E                    W16:                                   ; BAD-LEADER
F92E 4E                        DEC     SI                      ; CHECK RETRIES
F92F 7403                      JZ      W17                     ; JUMP IF TOO MANY RETRIES
F931 E965FF                    JMP     W4                      ; JUMP IF NOT TOO MANY RETRIES
F934                    W17:                                   ; NO VALID DATA FOUND

                        ;----- NO DATA FROM CASSETTE ERROR, I.E. TIMEOUT

F934 5E                        POP     SI                      ; RESTORE REGS
F935 59                        POP     CX                      ; RESTORE REGS
F936 5B                        POP     BX
F937 2BD2                      SUB     DX,DX                   ; ZERO NUMBER OF BYTES READ
F939 B404                      MOV     AH,04H                  ; TIME OUT ERROR (NO LEADER)
F93B 50                        PUSH    AX
F93C                    W18:                                   ; MOT-OFF
```

```
LOC OBJ           SOURCE

F93C E421                 IN      AL, 021H               ; RE_ENABLE INTERRUPTS
F93E 24FE                 AND     AL, 0FFH- 1
F940 E621                 OUT     021H, AL
F942 E845FF               CALL    MOTOR_OFF              ; TURN OFF MOTOR
F945 58                   POP     AX                     ; RESTORE RETURN CODE
F946 80FC01               CMP     AH,01H                 ; SET CARRY IF ERROR (AH>0)
F949 F5                   CMC
F94A C3                   RET                            ; FINISHED
                  READ_BLOCK      ENDP
                  ;----------------------------------------
                  ; PURPOSE:                               :
                  ;       TO READ A BYTE FROM CASSETTE     :
                  ; ON EXIT                                :
                  ;       REG AL CONTAINS READ DATA BYTE   :
                  ;----------------------------------------
F94B              READ_BYTE       PROC    NEAR
F94B 53                   PUSH    BX                     ; SAVE REGS BX,CX
F94C 51                   PUSH    CX
F94D B108                 MOV     CL,8H                  ; SET BIT COUNTER FOR 8 BITS
F94F              W19:                                   ; BYTE-ASM
F94F 51                   PUSH    CX                     ; SAVE CX
                  ;-------------------------------
                  ; READ DATA BIT FROM CASSETTE   :
                  ;-------------------------------
F950 E82600               CALL    READ_HALF_BIT          ; READ ONE PULSE
F953 E320                 JCXZ    W21                    ; IF CX=0 THEN TIMEOUT
                                                         ; BECAUSE OF NO DATA TRANSITIONS
F955 53                   PUSH    BX                     ; SAVE 1ST HALF BIT'S
                                                         ; PULSE WIDTH (IN BX)
F956 E82000               CALL    READ_HALF_BIT          ; READ COMPLEMENTARY PULSE
F959 58                   POP     AX                     ; COMPUTE DATA BIT
F95A E319                 JCXZ    W21                    ; IF CX=0 THEN TIMEOUT DUE TO
                                                         ; NO DATA TRANSITIONS
F95C 03D8                 ADD     BX,AX                  ; PERIOD
F95E 81FBF006             CMP     BX, 06F0H              ; CHECK FOR ZERO BIT
F962 F5                   CMC                            ; CARRY IS SET IF ONE BIT
F963 9F                   LAHF                           ; SAVE CARRY IN AH
F964 59                   POP     CX                     ; RESTORE CX
                                                         ; NOTE:
                                                         ;  MS BIT OF BYTE IS READ FIRST.
                                                         ;  REG CH IS SHIFTED LEFT WITH
                                                         ;  CARRY BEING INSERTED INTO LS
                                                         ;   BIT OF CH.
                                                         ; AFTER ALL 8 BITS HAVE BEEN
                                                         ;   READ, THE MS BIT OF THE DATA BYTE
                                                         ;   WILL BE IN THE MS BIT OF REG CH
F965 D0D5                 RCL     CH,1                   ; ROTATE REG CH LEFT WITH CARRY TO
                                                         ;   LS BIT OF REG CH
F967 9E                   SAHF                           ; RESTORE CARRY FOR CRC ROUTINE
F968 E80900               CALL    CRC_GEN                ; GENERATE CRC FOR BIT
F96B FEC9                 DEC     CL                     ; LOOP TILL ALL 8 BITS OF DATA
                                                         ; ASSEMBLED IN REG CH
F96D 75E0                 JNZ     W19                    ; BYTE_ASM
F96F 8AC5                 MOV     AL,CH                  ; RETURN DATA BYTE IN REG AL
F971 F8                   CLC
F972              W20:                                   ; RD-BYT-EX
F972 59                   POP     CX                     ; RESTORE REGS CX,BX
F973 5B                   POP     BX
F974 C3                   RET                            ; FINISHED
F975              W21:                                   ; NO-DATA
F975 59                   POP     CX                     ; RESTORE CX
F976 F9                   STC                            ; INDICATE ERROR
F977 EBF9                 JMP     W20                    ; RD_BYT_EX
                  READ_BYTE       ENDP
                  ;-----------------------------------------------
                  ; PURPOSE:                                      :
                  ;       TO COMPUTE TIME TILL NEXT DATA          :
                  ;       TRANSITION (EDGE)                       :
                  ; ON ENTRY:                                     :
                  ;       EDGE_CNT CONTAINS LAST EDGE COUNT       :
                  ; ON EXIT:                                      :
                  ;       AX CONTAINS OLD LAST EDGE COUNT         :
                  ;       BX CONTAINS PULSE WIDTH (HALF BIT)      :
                  ;-----------------------------------------------
F979              READ_HALF_BIT   PROC    NEAR
F979 B96400               MOV     CX, 100                ; SET TIME TO WAIT FOR BIT
F97C 8A266B00             MOV     AH,LAST_VAL            ; GET PRESENT INPUT VALUE
```

```
LOC OBJ               SOURCE

F980                  W22:                                  ; RD-H-BIT
F980 E462                    IN      AL,PORT_C              ; INPUT DATA BIT
F982 2410                    AND     AL,010H                ; MASK OFF EXTRANEOUS BITS
F984 3AC4                    CMP     AL,AH                  ; SAME AS BEFORE?
F986 E1F8                    LOOPE   W22                    ; LOOP TILL IT CHANGES
F988 A26B00                  MOV     LAST_VAL,AL            ; UPDATE LAST_VAL WITH NEW VALUE
F98B B000                    MOV     AL,0                   ; READ TIMER'S COUNTER COMMAND
F98D E643                    OUT     TIM_CTL,AL             ; LATCH COUNTER
F98F 8B1E6700                MOV     BX,EDGE_CNT            ; BX GETS LAST EDGE COUNT
F993 E440                    IN      AL,TIMER0              ; GET LS BYTE
F995 8AE0                    MOV     AH,AL                  ; SAVE IN AH
F997 E440                    IN      AL,TIMER0              ; GET MS BYTE
F999 86C4                    XCHG    AL,AH                  ; XCNG AL,AH
F99B 2BD8                    SUB     BX,AX                  ; SET BX EQUAL TO HALF BIT PERIOD
F99D A36700                  MOV     EDGE_CNT,AX            ; UPDATE EDGE COUNT;
F9A0 C3                      RET
                      READ_HALF_BIT   ENDP
                      ;--------------------------------------------------------------
                      ; PURPOSE                                                      :
                      ;       WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE.           :
                      ;       THE DATA IS PADDED TO FILL OUT THE LAST 256 BYTE BLOCK. :
                      ; ON ENTRY:                                                    :
                      ;       BX POINTS TO MEMORY BUFFER ADDRESS                     :
                      ;       CX CONTAINS NUMBER OF BYTES TO WRITE                   :
                      ; ON EXIT:                                                     :
                      ;       BX POINTS 1 BYTE PAST LAST BYTE WRITTEN TO CASSETTE    :
                      ;       CX IS ZERO                                             :
                      ;--------------------------------------------------------------
F9A1                  WRITE_BLOCK     PROC    NEAR
F9A1 53                      PUSH    BX
F9A2 51                      PUSH    CX
F9A3 E461                    IN      AL,PORT_B              ; DISABLE SPEAKER
F9A5 24FD                    AND     AL,NOT 02H
F9A7 0C01                    OR      AL, 01H                ; ENABLE TIMER
F9A9 E661                    OUT     PORT_B,AL
F9AB B0B6                    MOV     AL,0B6H                ; SET UP TIMER -- MODE 3 SQUARE WAVE
F9AD E643                    OUT     TIM_CTL,AL
F9AF E8A600                  CALL    BEGIN_OP               ; START MOTOR AND DELAY
F9B2 B8A004                  MOV     AX,1184                ; SET NORMAL BIT SIZE
F9B5 E88500                  CALL    W31                    ; SET_TIMER
F9B8 B90008                  MOV     CX,0800H               ; SET CX FOR LEADER BYTE COUNT
F9BB                  W23:                                  ; WRITE LEADER
F9BB F9                      STC                            ; WRITE ONE BITS
F9BC E86800                  CALL    WRITE_BIT
F9BF E2FA                    LOOP    W23                    ; LOOP 'TIL LEADER IS WRITTEN
F9C1 F8                      CLC                            ; WRITE SYNC BIT (0)
F9C2 E86200                  CALL    WRITE_BIT
F9C5 59                      POP     CX                     ; RESTORE REGS CX,BX
F9C6 5B                      POP     BX
F9C7 B016                    MOV     AL, 16H                ; WRITE SYN CHARACTER
F9C9 E84400                  CALL    WRITE_BYTE
                      ;--------------------------------------------------------------
                      ; PURPOSE                                                      :
                      ;       WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE            :
                      ; ON ENTRY:                                                    :
                      ;       BX POINTS TO MEMORY BUFFER ADDRESS                     :
                      ;       CONTAINS NUMBER OF BYTES TO WRITE                      :
                      ; ON EXIT:                                                     :
                      ;       BX POINTS 1 BYTE PAST LAST BYTE WRITTEN TO CASSETTE    :
                      ;       CX IS ZERO                                             :
                      ;--------------------------------------------------------------
F9CC                  WR_BLOCK:
F9CC C7066900FFFF            MOV     CRC_REG,0FFFFH         ; INIT CRC
F9D2 BA0001                  MOV     DX,256                 ; FOR 256 BYTES
F9D5                  W24:                                  ; WR-BLK
F9D5 268A07                  MOV     AL,ES:[BX]             ; READ BYTE FROM MEM
F9D8 E83500                  CALL    WRITE_BYTE             ; WRITE IT TO CASSETTE
F9DB E302                    JCXZ    W25                    ; UNLESS CX=0, ADVANCE PTRS & DEC COUNT
F9DD 43                      INC     BX                     ; INC BUFFER POINTER
F9DE 49                      DEC     CX                     ; DEC BYTE COUNTER
F9DF                  W25:                                  ; SKIP-ADV
F9DF 4A                      DEC     DX                     ; DEC BLOCK CNT
F9E0 7FF3                    JG      W24                    ; LOOP TILL 256 BYTE BLOCK
                                                            ;  IS WRITTEN TO TAPE
                      ;--------------------------------------------------------------
                      ; WRITE CRC                                                    :
                      ;       WRITE 1'S COMPLEMENT OF CRC REG TO CASSETTE            :
```

```
LOC OBJ              SOURCE

                     ;          WHICN IS CHECKED FOR CORRECTNESS WHEN THE BLOCK IS READ :
                     ; REG AX IS MODIFIED                                              :
                     ;-----------------------------------------------------------------
F9E2 A16900                   MOV     AX,CRC_REG              ; WRITE THE ONE'S COMPLEMENT OF THE
                                                              ;  TWO BYTE CRC TO TAPE
F9E5 F7D0                     NOT     AX                      ; FOR 1'S COMPLEMENT
F9E7 50                       PUSH    AX                      ; SAVE IT
F9E8 86E0                     XCHG    AH,AL                   ; WRITE MS BYTE FIRST
F9EA E82300                   CALL    WRITE_BTTE              ; WRITE IT
F9ED 58                       POP     AX                      ; GET IT BACK
F9EE E81F00                   CALL    WRITE_BTTE              ; NOW WRITE LS BYTE
F9F1 0BC9                     OR      CX,CX                   ; IS BYTE COUNT EXHAUSTED?
F9F3 75D7                     JNZ     WR_BLOCK                ; JUMP IF NOT DONE YET
F9F5 51                       PUSH    CX                      ; SAVE REG CX
F9F6 B92000                   MOV     CX, 32                  ; WRITE OUT TRAILER BITS
F9F9                 W26:                                     ; TRAIL-LOOP
F9F9 F9                       STC
F9FA E82A00                   CALL    WRITE_BIT
F9FD E2FA                     LOOP    W26                     ; WRITE UNTIL TRAILER WRITTEN
F9FF 59                       POP     CX                      ; RESTORE REG CX
FA00 B0B0                     MOV     AL, 0B0H                ; TURN TIMER2 OFF
FA02 E643                     OUT     TIM_CTL, AL
FA04 B80100                   MOV     AX, 1
FA07 E83300                   CALL    W31                     ; SET_TIMER
FA0A E87DFE                   CALL    MOTOR_OFF               ; TURN MOTOR OFF
FA0D 2BC0                     SUB     AX,AX                   ; NO ERRORS REPORTED ON WRITE OP
FA0F C3                       RET                             ; FINISNED
                     WRITE_BLOCK      ENDP
                     ;--------------------------------
                     ; WRITE A BYTE TO CASSETTE.     :
                     ; BYTE TO WRITE IS IN REG AL.   :
                     ;--------------------------------
FA10                 WRITE_BYTE       PROC    NEAR
FA10 51                       PUSH    CX                      ; SAVE REGS CX,AX
FA11 50                       PUSH    AX
FA12 8AE8                     MOV     CH,AL                   ; AL=BYTE TO WRITE.
                                                              ;   (MS BIT WRITTEN FIRST)
FA14 B108                     MOV     CL,8                    ; FOR 8 DATA BITS IN BYTE.
                                                              ;   NOTE: TWO EDGES PER BIT
FA16                 W27:                                     ; DISASSEMBLE THE DATA BIT
FA16 D0D5                     RCL     CH,1                    ; ROTATE MS BIT INTO CARRY
FA18 9C                       PUSHF                           ; SAVE FLAGS.
                                                              ;   NOTE: DATA BIT IS IN CARRY
FA19 E80B00                   CALL    WRITE_BIT               ; WRITE DATA BIT
FA1C 9D                       POPF                            ; RESTORE CARRY FOR CRC CALC
FA1D E82400                   CALL    CRC_GEN                 ; COMPUTE CRC ON DATA BIT
FA20 FEC9                     DEC     CL                      ; LOOP TILL ALL 8 BITS DONE
FA22 75F2                     JNZ     W27                     ; JUMP IF NOT DONE YET
FA24 58                       POP     AX                      ; RESTORE REGS AX,CX
FA25 59                       POP     CX
FA26 C3                       RET                             ; WE ARE FINISHED
                     WRITE_BYTE       ENDP
                     ;-----------------------------------------------------------
                     ; PURPOSE:                                                 :
                     ;          TO WRITE A DATA BIT TO CASSETTE                 :
                     ;          CARRY FLAG CONTAINS DATA BIT                    :
                     ;          I.E. IF SET DATA BIT IS A ONE                   :
                     ;          IF CLEAR DATA BIT IS A ZERO                     :
                     ;                                                          :
                     ;   NOTE: TWO EDGES ARE WRITTEN PER BIT                    :
                     ;          ONE BIT  HAS 500 USEC BETWEEN EDGES             :
                     ;               FOR A 1000 USEC PERIOD (1 MILLISEC)        :
                     ;                                                          :
                     ;          ZERO BIT HAS 250 USEC BETWEEN EDGES             :
                     ;               FOR A  500 USEC PERIOD (.5 MILLISEC)       :
                     ; CARRT FLAG IS DATA BIT                                   :
                     ;-----------------------------------------------------------
FA27                 WRITE_BIT        PROC    NEAR
                                                              ; ASSUME IT'S A '1'
FA27 B8A004                   MOV     AX,1184                 ; SET AX TO NOMINAL ONE SIZE
FA2A 7203                     JC      W28                     ; JUMP IF ONE BIT
FA2C B85002                   MOV     AX,592                  ; NO, SET TO NOMINAL ZERO SIZE
FA2F                 W28:                                     ; WRITE-BIT-AX
FA2F 50                       PUSH    AX                      ; WRITE BIT WITH PERIOD EQ TO VALUE AX
FA30                 W29:
FA30 E462                     IN      AL,PORT_C               ; INPUT TIMER_0 OUTPUT
FA32 2420                     AND     AL,020H
```

Appendix A

```
LOC OBJ                 SOURCE

FA34 74FA                       JZ      W29                     ; LOOP TILL HIGH
FA36                    W30:
FA36 E462                       IN      AL,PORT_C               ; NOW WAIT TILL TIMER'S OUTPUT IS LOW
FA38 2420                       AND     AL,020H
FA3A 75FA                       JNZ     W30
                                                                ; RELOAD TIMER WITH PERIOD
                                                                ; FOR NEXT DATA BIT
FA3C 58                         POP     AX                      ; RESTORE PERIOD COUNT
FA3D                    W31:                                    ; SET TIMER
FA3D E642                       OUT     042H, AL                ; SET LOW BYTE OF TIMER 2
FA3F 8AC4                       MOV     AL, AH
FA41 E642                       OUT     042H, AL                ; SET HIGH BYTE OF TIMER 2
FA43 C3                         RET
                        WRITE_BIT       ENDP
                        ;----------------------------------------------
                        ;  UPDATE CRC REGISTER WITH NEXT DATA BIT      :
                        ;  CRC IS USED TO DETECT READ ERRORS           :
                        ;  ASSUMES DATA BIT IS IN CARRY                :
                        ;                                              :
                        ;  REG AX IS MODIFIED                          :
                        ;  FLAGS ARE MODIFIED                          :
                        ;----------------------------------------------
FA44                    CRC_GEN         PROC    NEAR
FA44 A16900                     MOV     AX,CRC_REG
                                                                ; THE FOLLOWING INSTUCTIONS
                                                                ; WILL SET THE OVERFLOW FLAG
                                                                ; IF CARRY AND MS BIT OF CRC
                                                                ; ARE UNEQUAL
FA47 D1D8                       RCR     AX,1
FA49 D1D0                       RCL     AX,1
FA4B F8                         CLC                             ; CLEAR CARRY
FA4C 7104                       JNO     W32                     ; SKIP IF NO OVERFLOW
                                                                ; IF DATA BIT XORED WITH
                                                                ;  CRC REG BIT 15 IS ONE
FA4E 351008                     XOR     AX,0810H                ; THEN XOR CRC REG WITH 0801H
FA51 F9                         STC                             ; SET CARRY
FA52                    W32:
FA52 D1D0                       RCL     AX,1                    ; ROTATE CARRY (DATA BIT)
                                                                ; INTO CRC REG
FA54 A36900                     MOV     CRC_REG,AX              ; UPDATE CRC_REG
FA57 C3                         RET                             ; FINISHED
                        CRC_GEN         ENDP

FA58                    BEGIN_OP        PROC    NEAR            ; START TAPE AND DELAY
FA58 E826FE                     CALL    MOTOR_ON                ;TURN ON MOTOR
FA5B B342                       MOV     BL,42H                  ;DELAY FOR TAPE DRIVE
                                                                ;TO GET UP TO SPEED  (1/2 SEC)
FA5D                    W33:
FA5D B90007                     MOV     CX,700H                 ;INNER LOOP= APPROX. 10 MILLISEC
FA60 E2FE               W34:    LOOP    W34
FA62 FECB                       DEC     BL
FA64 75F7                       JNZ     W33
FA66 C3                         RET
                        BEGIN_OP        ENDP

FA67 20323031           E1      DB      ' 201',13,10
FA6B 0D
FA6C 0A

                        ;----------------------------------------------------------------
                        ;       CHARACTER GENERATOR GRAPHICS FOR 320X200 AND 640X200 GRAPHICS
                        ;----------------------------------------------------------------
FA6E                            ORG     0FA6EH
FA6E                    CRT_CHAR_GEN    LABEL   BYTE
FA6E 0000000000000000           DB      000H,000H,000H,000H,000H,000H,000H,000H ; D_00
FA76 7E81A581BD99817E           DB      07EH,081H,0A5H,081H,0BDH,099H,081H,07EH ; D_01
FA7E 7EFFDBFFC3E7FF7E           DB      07EH,0FFH,0DBH,0FFH,0C3H,0E7H,0FFH,07EH ; D_02
FA86 6CFEFEFE7C381000           DB      06CH,0FEH,0FEH,0FEH,07CH,038H,010H,000H ; D_03
FA8E 10387CFE7C381000           DB      010H,038H,07CH,0FEH,07CH,038H,010H,000H ; D_04
FA96 387C38FEFE7C387C           DB      038H,07CH,038H,0FEH,0FEH,07CH,038H,07CH ; D_05
FA9E 1010387CFE7C387C           DB      010H,010H,038H,07CH,0FEH,07CH,038H,07CH ; D_06
FAA6 0000183C3C180000           DB      000H,000H,018H,03CH,03CH,018H,000H,000H ; D_07
FAAE FFFFE7C3C3E7FFFF           DB      0FFH,0FFH,0E7H,0C3H,0C3H,0E7H,0FFH,0FFH ; D_08
FAB6 003C664242663C00           DB      000H,03CH,066H,042H,042H,066H,03CH,000H ; D_09
FABE FFC399BDBD99C3FF           DB      0FFH,0C3H,099H,0BDH,0BDH,099H,0C3H,0FFH ; D_0A
FAC6 0F070F7DCCCCCC78           DB      00FH,007H,00FH,07DH,0CCH,0CCH,0CCH,078H ; D_0B
FACE 3C6666663C187E18           DB      03CH,066H,066H,066H,03CH,018H,07EH,018H ; D_0C
```

```
LOC OBJ                   SOURCE

FAD6 3F333F303070F0E0     DB   03FH,033H,03FH,030H,030H,070H,0F0H,0E0H ; D_0D
FADE 7F637F636367E6C0     DB   07FH,063H,07FH,063H,063H,067H,0E6H,0C0H ; D_0E
FAE6 995A3CE7E73C5A99     DB   099H,05AH,03CH,0E7H,0E7H,03CH,05AH,099H ; D_0F
FAEE 80E0F8FEF8E08000     DB   080H,0E0H,0F8H,0FEH,0F8H,0E0H,080H,000H ; D_10
FAF6 020E3EFE3E0E0200     DB   002H,00EH,03EH,0FEH,03EH,00EH,002H,000H ; D_11
FAFE 183C7E18187E3C18     DB   018H,03CH,07EH,018H,018H,07EH,03CH,018H ; D_12
FB06 6666666666006600     DB   066H,066H,066H,066H,066H,000H,066H,000H ; D_13
FB0E 7FDBDB7B1B1B1B00     DB   07FH,0DBH,0DBH,07BH,01BH,01BH,01BH,000H ; D_14
FB16 3E63386C6C38CC78     DB   03EH,063H,038H,06CH,06CH,038H,0CCH,078H ; D_15
FB1E 000000007E7E7E00     DB   000H,000H,000H,000H,07EH,07EH,07EH,000H ; D_16
FB26 183C7E187E3C18FF     DB   018H,03CH,07EH,018H,07EH,03CH,018H,0FFH ; D_17
FB2E 183C7E1818181800     DB   018H,03CH,07EH,018H,018H,018H,018H,000H ; D_18
FB36 181818187E3C1800     DB   018H,018H,018H,018H,07EH,03CH,018H,000H ; D_19
FB3E 00180CFE0C180000     DB   000H,018H,00CH,0FEH,00CH,018H,000H,000H ; D_1A
FB46 003060FE60300000     DB   000H,030H,060H,0FEH,060H,030H,000H,000H ; D_1B
FB4E 0000C0C0C0FE0000     DB   000H,000H,0C0H,0C0H,0C0H,0FEH,000H,000H ; D_1C
FB56 002466FF66240000     DB   000H,024H,066H,0FFH,066H,024H,000H,000H ; D_1D
FB5E 00183C7EFFFF0000     DB   000H,018H,03CH,07EH,0FFH,0FFH,000H,000H ; D_1E
FB66 00FFFF7E3C180000     DB   000H,0FFH,0FFH,07EH,03CH,018H,000H,000H ; D_1F
FB6E 0000000000000000     DB   000H,000H,000H,000H,000H,000H,000H,000H ; SP D_20
FB76 3078783030003000     DB   030H,078H,078H,030H,030H,000H,030H,000H ; ! D_21
FB7E 6C6C6C0000000000     DB   06CH,06CH,06CH,000H,000H,000H,000H,000H ; " D_22
FB86 6C6CFE6CFE6C6C00     DB   06CH,06CH,0FEH,06CH,0FEH,06CH,06CH,000H ; # D_23
FB8E 307CC0780CF83000     DB   030H,07CH,0C0H,078H,00CH,0F8H,030H,000H ; $ D_24
FB96 00C6CC183066C600     DB   000H,0C6H,0CCH,018H,030H,066H,0C6H,000H ; PER CENT D_25
FB9E 386C3876DCCC7600     DB   038H,06CH,038H,076H,0DCH,0CCH,076H,000H ; & D_26
FBA6 6060C00000000000     DB   060H,060H,0C0H,000H,000H,000H,000H,000H ; ' D_27
FBAE 1830606060301800     DB   018H,030H,060H,060H,060H,030H,018H,000H ; ( D_28
FBB6 6030181818306000     DB   060H,030H,018H,018H,018H,030H,060H,000H ; ) D_29
FBBE 00663CFF3C660000     DB   000H,066H,03CH,0FFH,03CH,066H,000H,000H ; * D_2A
FBC6 003030FC30300000     DB   000H,030H,030H,0FCH,030H,030H,000H,000H ; + D_2B
FBCE 0000000000303060     DB   000H,000H,000H,000H,000H,030H,030H,060H ; , D_2C
FBD6 000000FC00000000     DB   000H,000H,000H,0FCH,000H,000H,000H,000H ; - D_2D
FBDE 0000000000303000     DB   000H,000H,000H,000H,000H,030H,030H,000H ; . D_2E
FBE6 060C183060C08000     DB   006H,00CH,018H,030H,060H,0C0H,080H,000H ; / D_2F
FBEE 7CC6CEDEF6E67C00     DB   07CH,0C6H,0CEH,0DEH,0F6H,0E6H,07CH,000H ; 0 D_30
FBF6 307030303030FC00     DB   030H,070H,030H,030H,030H,030H,0FCH,000H ; 1 D_31
FBFE 78CC0C3860CCFC00     DB   078H,0CCH,00CH,038H,060H,0CCH,0FCH,000H ; 2 D_32
FC06 78CC0C380CCC7800     DB   078H,0CCH,00CH,038H,00CH,0CCH,078H,000H ; 3 D_33
FC0E 1C3C6CCCFE0C1E00     DB   01CH,03CH,06CH,0CCH,0FEH,00CH,01EH,000H ; 4 D_34
FC16 FCC0F80C0CCC7800     DB   0FCH,0C0H,0F8H,00CH,00CH,0CCH,078H,000H ; 5 D_35
FC1E 3860C0F8CCCC7800     DB   038H,060H,0C0H,0F8H,0CCH,0CCH,078H,000H ; 6 D_36
FC26 FCCC0C1830303000     DB   0FCH,0CCH,00CH,018H,030H,030H,030H,000H ; 7 D_37
FC2E 78CCCC78CCCC7800     DB   078H,0CCH,0CCH,078H,0CCH,0CCH,078H,000H ; 8 D_38
FC36 78CCCC7C0C187000     DB   078H,0CCH,0CCH,07CH,00CH,018H,070H,000H ; 9 D_39
FC3E 0030300000303000     DB   000H,030H,030H,000H,000H,030H,030H,000H ; : D_3A
FC46 0030300000303060     DB   000H,030H,030H,000H,000H,030H,030H,060H ; ; D_3B
FC4E 183060C060301800     DB   018H,030H,060H,0C0H,060H,030H,018H,000H ; < D_3C
FC56 0000FC0000FC0000     DB   000H,000H,0FCH,000H,000H,0FCH,000H,000H ; = D_3D
FC5E 6030180C18306000     DB   060H,030H,018H,00CH,018H,030H,060H,000H ; > D_3E
FC66 78CC0C1830003000     DB   078H,0CCH,00CH,018H,030H,000H,030H,000H ; ? D_3F
FC6E 7CC6DEDEDEC07800     DB   07CH,0C6H,0DEH,0DEH,0DEH,0C0H,078H,000H ; @ D_40
FC76 3078CCCCFCCCCC00     DB   030H,078H,0CCH,0CCH,0FCH,0CCH,0CCH,000H ; A D_41
FC7E FC66667C6666FC00     DB   0FCH,066H,066H,07CH,066H,066H,0FCH,000H ; B D_42
FC86 3C66C0C0C0663C00     DB   03CH,066H,0C0H,0C0H,0C0H,066H,03CH,000H ; C D_43
FC8E F86C6666666CF800     DB   0F8H,06CH,066H,066H,066H,06CH,0F8H,000H ; D D_44
FC96 FE6268786862FE00     DB   0FEH,062H,068H,078H,068H,062H,0FEH,000H ; E D_45
FC9E FE6268786860F000     DB   0FEH,062H,068H,078H,068H,060H,0F0H,000H ; F D_46
FCA6 3C66C0C0CE663E00     DB   03CH,066H,0C0H,0C0H,0CEH,066H,03EH,000H ; G D_47
FCAE CCCCCCFCCCCCCC00     DB   0CCH,0CCH,0CCH,0FCH,0CCH,0CCH,0CCH,000H ; H D_48
FCB6 7830303030307800     DB   078H,030H,030H,030H,030H,030H,078H,000H ; I D_49
FCBE 1E0C0C0CCCCC7800     DB   01EH,00CH,00CH,00CH,0CCH,0CCH,078H,000H ; J D_4A
FCC6 E6666C786C66E600     DB   0E6H,066H,06CH,078H,06CH,066H,0E6H,000H ; K D_4B
FCCE F06060606266FE00     DB   0F0H,060H,060H,060H,062H,066H,0FEH,000H ; L D_4C
FCD6 C6EEFEFED6C6C600     DB   0C6H,0EEH,0FEH,0FEH,0D6H,0C6H,0C6H,000H ; M D_4D
FCDE C6E6F6DECEC6C600     DB   0C6H,0E6H,0F6H,0DEH,0CEH,0C6H,0C6H,000H ; N D_4E
FCE6 386CC6C6C66C3800     DB   038H,06CH,0C6H,0C6H,0C6H,06CH,038H,000H ; O D_4F
FCEE FC66667C6060F000     DB   0FCH,066H,066H,07CH,060H,060H,0F0H,000H ; P D_50
FCF6 78CCCCCCDC781C00     DB   078H,0CCH,0CCH,0CCH,0DCH,078H,01CH,000H ; Q D_51
FCFE FC66667C6C66E600     DB   0FCH,066H,066H,07CH,06CH,066H,0E6H,000H ; R D_52
FD06 78CCE0701CCC7800     DB   078H,0CCH,0E0H,070H,01CH,0CCH,078H,000H ; S D_53
FD0E FCB4303030307800     DB   0FCH,0B4H,030H,030H,030H,030H,078H,000H ; T D_54
FD16 CCCCCCCCCCCCFC00     DB   0CCH,0CCH,0CCH,0CCH,0CCH,0CCH,0FCH,000H ; U D_55
FD1E CCCCCCCCCC783000     DB   0CCH,0CCH,0CCH,0CCH,0CCH,078H,030H,000H ; V D_56
FD26 C6C6C6D6FEEEC600     DB   0C6H,0C6H,0C6H,0D6H,0FEH,0EEH,0C6H,000H ; W D_57
FD2E C6C66C38386CC600     DB   0C6H,0C6H,06CH,038H,038H,06CH,0C6H,000H ; X D_58
FD36 CCCCCC7830307800     DB   0CCH,0CCH,0CCH,078H,030H,030H,078H,000H ; Y D_59
```

Appendix A

```
LOC OBJ                      SOURCE

FD3E FEC68C183266FE00              DB      0FEH,0C6H,08CH,018H,032H,066H,0FEH,000H ; Z D_5A
FD46 7860606060607800              DB      078H,060H,060H,060H,060H,060H,078H,000H ; [ D_5B
FD4E C06030180C060200              DB      0C0H,060H,030H,018H,00CH,006H,002H,000H ; BACKSLASH D_5C
FD56 7818181818187800              DB      078H,018H,018H,018H,018H,018H,078H,000H ; ] D_5D
FD5E 10386CC600000000              DB      010H,038H,06CH,0C6H,000H,000H,000H,000H ; CIRCUMFLEX D_5E
FD66 00000000000000FF              DB      000H,000H,000H,000H,000H,000H,000H,0FFH ; _ D_5F
FD6E 3030180000000000              DB      030H,030H,018H,000H,000H,000H,000H,000H ; ' D_60
FD76 0000780C7CCC7600              DB      000H,000H,078H,00CH,07CH,0CCH,076H,000H ; LOWER CASE A D_61
FD7E E060607C6666DC00              DB      0E0H,060H,060H,07CH,066H,066H,0DCH,000H ; L.C. B D_62
FD86 000078CCC0CC7800              DB      000H,000H,078H,0CCH,0C0H,0CCH,078H,000H ; L.C. C D_63
FD8E 1C0C0C7CCCCC7600              DB      01CH,00CH,00CH,07CH,0CCH,0CCH,076H,000H ; L.C. D D_64
FD96 000078CCFCC07800              DB      000H,000H,078H,0CCH,0FCH,0C0H,078H,000H ; L.C. E D_65
FD9E 386C60F06060F000              DB      038H,06CH,060H,0F0H,060H,060H,0F0H,000H ; L.C. F D_66
FDA6 000076CCCC7C0CF8              DB      000H,000H,076H,0CCH,0CCH,07CH,00CH,0F8H ; L.C. G D_67
FDAE E0606C766666E600              DB      0E0H,060H,06CH,076H,066H,066H,0E6H,000H ; L.C. H D_68
FDB6 3000703030307800              DB      030H,000H,070H,030H,030H,030H,078H,000H ; L.C. I D_69
FDBE 0C000C0C0CCCCC78              DB      00CH,000H,00CH,00CH,00CH,0CCH,0CCH,078H ; L.C. J D_6A
FDC6 E060666C786CE600              DB      0E0H,060H,066H,06CH,078H,06CH,0E6H,000H ; L.C. K D_6B
FDCE 7030303030307800              DB      070H,030H,030H,030H,030H,030H,078H,000H ; L.C. L D_6C
FDD6 0000CCFEFED6C600              DB      000H,000H,0CCH,0FEH,0FEH,0D6H,0C6H,000H ; L.C. M D_6D
FDDE 0000F8CCCCCCCC00              DB      000H,000H,0F8H,0CCH,0CCH,0CCH,0CCH,000H ; L.C. H D_6E
FDE6 000078CCCCCC7800              DB      000H,000H,078H,0CCH,0CCH,0CCH,078H,000H ; L.C. O D_6F
FDEE 0000DC66667C60F0              DB      000H,000H,0DCH,066H,066H,07CH,060H,0F0H ; L.C. P D_70
FDF6 000076CCCC7C0C1E              DB      000H,000H,076H,0CCH,0CCH,07CH,00CH,01EH ; L.C. Q D_71
FDFE 0000DC766660F000              DB      000H,000H,0DCH,076H,066H,060H,0F0H,000H ; L.C. R D_72
FE06 00007CC0780CF800              DB      000H,000H,07CH,0C0H,078H,00CH,0F8H,000H ; L.C. S D_73
FE0E 10307C3030341800              DB      010H,030H,07CH,030H,030H,034H,018H,000H ; L.C. T D_74
FE16 0000CCCCCCCC7600              0B      000H,000H,0CCH,0CCH,0CCH,0CCH,076H,000H ; L.C. U D_75
FE1E 0000CCCCCC783000              0B      000H,000H,0CCH,0CCH,0CCH,078H,030H,000H ; L.C. V 0_76
FE26 0000C6D6FEFE6C00              DB      000H,000H,0C6H,0D6H,0FEH,0FEH,06CH,000H ; L.C. W D_77
FE2E 0000C66C386CC600              0B      000H,000H,0C6H,06CH,038H,06CH,0C6H,000H ; L.C. X 0_78
FE36 0000CCCCCC7C0CF8              DB      000H,000H,0CCH,0CCH,0CCH,07CH,00CH,0F8H ; L.C. Y D_79
FE3E 0000FC983064FC00              08      000H,000H,0FCH,098H,030H,064H,0FCH,000H ; L.C. Z 0_7A
FE46 1C3030E030301C00              0B      01CH,030H,030H,0E0H,030H,030H,01CH,000H ; { 0_7B
FE4E 1818180018181800              DB      018H,018H,018H,000H,018H,018H,018H,000H ; | 0_7C
FE56 E030301C3030E000              DB      0E0H,030H,030H,01CH,030H,030H,0E0H,000H ; } 0_7D
FE5E 76DC000000000000              0B      076H,0DCH,000H,000H,000H,000H,000H,000H ; TILDE D_7E
FE66 0010386CC6C6FE00              0B      000H,010H,038H,06CH,0C6H,0C6H,0FEH,000H ; DELTA 0_7F

                             ;--- INT 1A -------------------------------------------
                             ; TIME_OF_0AY                                          :
                             ;  THIS ROUTINE ALLDWS THE CLOCK TO BE SET/REA0         :
                             ;                                                      :
                             ; INPUT                                                :
                             ;   (AH) = 0     REA0 THE CURRENT CLOCK SETTING        :
                             ;                RETURHS CX = HIGH PORTION OF COUNT    :
                             ;                        OX = LOW PORTION OF COUNT     :
                             ;                        AL = 0 IF TIMER HAS NOT PASSE0 :
                             ;                         24 HOURS SINCE LAST REA0     :
                             ;                           <>0 IF ON ANOTHER 0AY      :
                             ;   (AH) = 1     SET THE CURRENT CLOCK                 :
                             ;        CX = HIGH PORTION OF COUNT                    :
                             ;        0X = LDW PORTION OF CDUNT                     :
                             ; HOTE: COUNTS OCCUR AT THE RATE OF                    :
                             ;         1193180/65536 COUNTS/SEC                     :
                             ;       (OR ABDUT 18.2 PER SECOND -- SEE EQUATES BELOW) :
                             ;-------------------------------------------------------
                                   ASSUME  CS:CDDE,DS:0ATA
FE6E                               DRG     0FE6EH
FE6E                         TIME_DF_DAY     PRDC    FAR
FE6E FB                            STI                             ; INTERRUPTS BACK OH
FE6F 1E                            PUSH    DS                      ; SAVE SEGMENT
FE70 E8CB00                        CALL    DDS
FE73 0AE4                          DR      AH,AH                   ; AH=0
FE75 7407                          JZ      T2                      ; READ_TIME
FE77 FECC                          DEC     AH                      ; AH=1
FE79 7416                          JZ      T3                      ; SET_TIME
FE7B                         T1:                                   ; TOD_RETURN
FE7B FB                            STI                             ; INTERRUPTS BACK ON
FE7C 1F                            PDP     DS                      ; RECOVER SEGMENT
FE7D CF                            IRET                            ; RETURN TD CALLER
FE7E                         T2:                                   ; READ_TIME
FE7E FA                            CLI                             ; NO TIMER INTERRUPTS WHILE READING
FE7F A07000                        MDV     AL,TIMER_DFL
FE82 C606700000                    MDV     TIMER_DFL,0             ; GET OVERFLOW, AND RESET THE FLAG
FE87 8B0E6E00                      MDV     CX,TIMER_HIGH
FE8B 8B166C00                      MDV     DX,TIMER_LOW
```

```
LOC OBJ                 SOURCE

FE8F EBEA                      JMP     T1                      ; TOD_RETURN
FE91                    T3:                                    ; SET_TIME
FE91 FA                        CLI                             ; NO INTERRUPTS WHILE WRITING
FE92 89166C00                  MOV     TIMER_LOW,OX
FE96 890E6E00                  MOV     TIMER_HIGH,CX           ; SET THE TIME
FE9A C606700000                MOV     TIMER_OFL,0             ; RESET OVERFLDW
FE9F EBDA                      JMP     T1                      ; TOD_RETURN
                        TIME_OF_OAY    ENDP

                        ;-------------------------------------------------------
                        ; THIS ROUTINE HANOLES THE TIMER INTERRUPT FROM          :
                        ;  CNANNEL 0 DF THE 8253 TIMER. INPUT FREQUENCY          :
                        ;  IS 1.19318 MHZ AND THE DIVISOR IS 65536, RESULTING    :
                        ;  IN APPROX. 18.2 INTERRUPTS EVERY SECOND.              :
                        ;                                                        :
                        ; THE INTERRUPT HANDLER MAINTAINS A COUNT OF INTERRUPTS  :
                        ;  SINCE POWER ON TIME, WHICH MAY BE USED TD ESTABLISH   :
                        ;  TIME OF OAY.                                          :
                        ; THE INTERRUPT HANDLER ALSO OECREMENTS THE MDTOR        :
                        ;  CONTROL COUNT OF THE DISKETTE, AND WHEN IT EXPIRES,   :
                        ;  WILL TURN OFF THE DISKETTE MOTOR, AND RESET THE       :
                        ;  MOTOR RUNNING FLAGS.                                  :
                        ; THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE  :
                        ;  THROUGH INTERRUPT ICH AT EVERY TIME TICK.  THE USER   :
                        ;  MUST CODE A ROUTINE AND PLACE THE CORRECT ADDRESS IN  :
                        ;  THE VECTOR TABLE.                                     :
                        ;-------------------------------------------------------
FEA5                           ORG     OFEA5H
FEA5                    TIMER_INT      PROC    FAR
FEA5 FB                        STI                             ; INTERRUPTS BACK ON
FEA6 1E                        PUSH    OS
FEA7 50                        PUSH    AX
FEA8 52                        PUSH    DX                      ; SAVE MACHINE STATE
FEA9 E89200                    CALL    DDS
FEAC FF066C00                  INC     TIMER_LOW               ; INCREMENT TIME
FEB0 7504                      JNZ     T4                      ; TEST_DAY
FEB2 FF066E00                  INC     TIMER_HIGH              ; INCREMENT HIGH WORD DF TIME
FEB6                    T4:                                    ; TEST_DAY
FEB6 833E6E0018                CMP     TIMER_HIGH,018H         ; TEST FOR CDUNT EQUALING 24 HOURS
FEBB 7515                      JNZ     T5                      ; OISKETTE_CTL
FEBD 813E6C00B000              CMP     TIMER_LOW,0B0H
FEC3 750D                      JNZ     T5                      ; DISKETTE_CTL

                        ;------ TIMER HAS GONE 24 HOURS

FEC5 2BC0                      SUB     AX,AX
FEC7 A36E00                    MDV     TIMER_HIGH,AX
FECA A36C00                    MDV     TIMER_LOW,AX
FECD C606700001                MOV     TIMER_DFL,1

                        ;------ TEST FOR DISKETTE TIME OUT

FED2                    T5:                                    ; DISKETTE_CTL
FED2 FE0E4000                  DEC     MOTOR_COUNT
FED6 750B                      JNZ     T6                      ; RETURN IF COUNT NOT OUT
FED8 80263F00F0                AND     MOTOR_STATUS,0F0H       ; TURN OFF MDTOR RUNNING BITS
FEDD B00C                      MDV     AL,0CH
FEDF BAF203                    MOV     DX,03F2H                ; FDC CTL PORT
FEE2 EE                        OUT     DX,AL                   ; TURN OFF THE MOTOR
FEE3                    T6:                                    ; TIMER_RET:
FEE3 CD1C                      INT     1CH                     ; TRANSFER CONTROL TO A USER ROUTINE
FEE5 B020                      MOV     AL,EDI
FEE7 E620                      OUT     020H,AL                 ; END OF INTERRUPT TO 8259
FEE9 5A                        POP     OX
FEEA 58                        POP     AX
FEEB 1F                        POP     OS                      ; RESET MACHINE STATE
FEEC CF                        IRET                            ; RETURN FROM INTERRUPT
                        TIMER_INT      ENDP

FEED 31383031           F3B    OB      '1801',13,10
FEF1 00
FEF2 0A

                        ;--------------------------------------------------------------
                        ;       THESE ARE THE VECTORS WHICH ARE MOVEO INTO            :
                        ;       THE 8086 INTERRUPT AREA OURING POWER ON.              :
                        ;       ONLT THE DFFSETS ARE DISPLATED HERE, CODE SEGMENT     :
```

Appendix A

```
LOC OBJ              LINE    SOURCE

                             ;       WILL BE ADDED FOR ALL OF THEM, EXCEPT WHERE NOTED       :
                             ;----------------------------------------------------------------
                                     ASSUME  CS:CODE
FEF3                                 ORG     0FEF3H
FEF3                         VECTOR_TABLE    LABEL   WORD        ; VECTOR TABLE FOR MOVE TO INTERRUPTS
FEF3 A5FE                            DW      OFFSET TIMER_INT    ; INTERRUPT 8
FEF5 87E9                            DW      OFFSET  KB_INT      ; INTERRUPT 9
FEF7 DDE6                            DW      OFFSET D_EOI        ; INTERRUPT A
FEF9 DDE6                            DW      OFFSET D_EOI        ; INTERRUPT B
FEFB DDE6                            DW      OFFSET D_EOI        ; INTERRUPT C
FEFD DDE6                            DW      OFFSET D_EOI        ; INTERRUPT D
FEFF 57EF                            DW      OFFSET DISK_INT     ; INTERRUPT E
FF01 DDE6                            DW      OFFSET D_EOI        ; INTERRUPT F
FF03 65F0                            DW      OFFSET VIDEO_IO     ; INTERRUPT 10H
FF05 40F8                            DW      OFFSET EQUIPMENT    ; INTERRUPT 11H
FF07 41F8                            DW      OFFSET MEMORY_SIZE_DET  ; INTERRUPT 12H
FF09 59EC                            DW      OFFSET DISKETTE_IO  ; INTERRUPT 13H
FF0B 39E7                            DW      OFFSET RS232_IO     ; INTERRUPT 14H
FF0D 59F8                            DW      OFFSET CASSETTE_IO  ; INTERRUPT 15H
FF0F 2EE8                            DW      OFFSET KEYBOARD_IO  ; INTERRUPT 16H
FF11 D2EF                            DW      OFFSET PRINTER_IO   ; INTERRUPT 17H

FF13 0000                            DW      00000H              ; INTERRUPT 18H
                             ;       DW      0F600H              ;  MUST BE INSERTED INTO TABLE LATER

FF15 F2E6                            DW      OFFSET BOOT_STRAP   ; INTERRUPT 19H
FF17 6EFE                            DW      TIME_OF_DAY         ; INTERRUPT 1AH -- TIME OF DAY
FF19 53FF                            DW      DUMMY_RETURN        ; INTERRUPT 1BH -- KEYBOARD BREAK ADDR
FF1B 53FF                            DW      DUMMY_RETURN        ; INTERRUPT 1C -- TIMER BREAK ADDR
FF1D A4F0                            DW      VIDEO_PARMS         ; INTERRUPT 1D -- VIDEO PARAMETERS
FF1F C7EF                            DW      OFFSET DISK_BASE    ; INTERRUPT 1E -- DISK PARMS
FF21 0000                            DW      0                   ; INTERRUPT 1F -- POINTER TO VIDEO EXT

FF23 50415249545920          D2      DB      'PARITY CHECK 1',13,10
     434845434B2031
FF31 0D
FF32 0A
FF33 20333031                F1      DB      ' 301',13,10
FF37 0D
FF38 0A
FF39 313331                  F2      DB      '131',13,10
FF3C 0D
FF3D 0A

FF3E                         DDS     PROC    NEAR
FF3E 50                              PUSH    AX                  ; SAVE AX
FF3F B84000                          MOV     AX,DATA
FF42 8ED8                            MOV     DS,AX               ; SET DATA SEGMENT
FF44 58                              POP     AX                  ; RESTORE AX
FF45 C3                              RET
                             DDS     ENDP

                             ;--------------------------------------------------------
                             ;       TEMPORARY INTERRUPT SERVICE ROUTINE             :
                             ;--------------------------------------------------------
FF47                                 ORG     0FF47H
FF47                         D11     PROC    NEAR
FF47 B401                            MOV     AH,1
FF49 50                              PUSH    AX                  ; SAVE REG AX CONTENTS
FF4A B0FF                            MOV     AL,0FFH             ; MASK ALL INTERRUPTS OFF
FF4C E621                            OUT     INTA01,AL
FF4E B020                            MOV     AL,EOI
FF50 E620                            OUT     INTA00,AL
FF52 58                              POP     AX                  ; RESTORE REG AX CONTENTS
FF53                         DUMMY_RETURN:                       ; NEED IRET FOR VECTOR TABLE
FF53 CF                              IRET
                             D11     ENDP

                             ;-- INT 5 -------------------------------------------------------
                             ;       THIS LOGIC WILL BE INVOKED BY INTERRUPT 05H TO PRINT THE      :
                             ;       SCREEN. THE CURSOR POSITION AT THE TIME THIS ROUTINE IS INVOKED :
                             ;       WILL BE SAVED AND RESTORED UPON COMPLETION. THE ROUTINE IS    :
                             ;       INTENDED TO RUN WITH INTERRUPTS ENABLED. IF A SUBSEQUENT      :
                             ;       'PRINT SCREEN' KEY IS DEPRESSED DURING THE TIME THIS ROUTINE  :
                             ;       IS PRINTING IT WILL BE IGNORED.                               :
                             ;       ADDRESS 50:0 CONTAINS THE STATUS OF THE PRINT SCREEN:         :
                             ;                                                                     :
```

```
LOC OBJ                 SOURCE

                        ;       50:0    =0      EITHER PRINT SCREEN HAS NOT BEEN CALLED     :
                        ;                       OR UPON RETURN FROM A CALL THIS INDICATES   :
                        ;                       A SUCCESSFUL OPERATION.                     :
                        ;               =1      PRINT SCREEN IS IN PROGRESS                 :
                        ;               =255    ERROR ENCOUNTERED DURING PRINTING           :
                        ;----------------------------------------------------------------------
                                ASSUME  CS:CODE,DS:XXDATA
FF54                            ORG     0FF54H
FF54                    PRINT_SCREEN    PROC    FAR
FF54 FB                         STI                             ; MUST RUN WITH INTERRUPTS ENABLED
FF55 1E                         PUSH    DS                      ; MUST USE 50:0 FOR DATA AREA STORAGE
FF56 50                         PUSH    AX
FF57 53                         PUSH    BX
FF58 51                         PUSH    CX                      ; WILL USE THIS LATER FOR CURSOR LIMITS
FF59 52                         PUSH    DX                      ; WILL HOLD CURRENT CURSOR POSITION
FF5A B85000                     MOV     AX,XXDATA               ; HEX 50
FF5D 8ED8                       MOV     DS,AX
FF5F 803E000001                 CMP     STATUS_BYTE,1           ; SEE IF PRINT ALREADY IN PROGRESS
FF64 745F                       JZ      EXIT                    ; JUMP IF PRINT ALREADY IN PROGRESS
FF66 C606000001                 MOV     STATUS_BYTE,1           ; INDICATE PRINT NOW IN PROGRESS
FF6B B40F                       MOV     AH,15                   ; WILL REQUEST THE CURRENT SCREEN MODE
FF6D CD10                       INT     10H                     ;       [AL]=MODE
                                                                ;       [AH]=NUMBER COLUMNS/LINE
                                                                ;       [BH]=VISUAL PAGE
                        ;-----------------------------------------------------------------
                        ;       AT THIS POINT WE KNOW THE COLUMNS/LINE ARE IN          :
                        ;       [AX] AND THE PAGE IF APPLICABLE IS IN [BH]. THE STACK  :
                        ;       HAS DS,AX,BX,CX,DX PUSHED. [AL] HAS VIDEO MODE         :
                        ;-----------------------------------------------------------------
FF6F 8ACC                       MOV     CL,AH                   ; WILL MAKE USE OF [CX] REGISTER TO
FF71 B519                       MOV     CH,25                   ; CONTROL ROW & COLUMNS
FF73 E85500                     CALL    CRLF                    ; CARRIAGE RETURN LINE FEED ROUTINE
FF76 51                         PUSH    CX                      ; SAVE SCREEN BOUNDS
FF77 B403                       MOV     AH,3                    ; WILL NOW READ THE CURSOR.
FF79 CD10                       INT     10H                     ; AND PRESERVE THE POSITION
FF7B 59                         POP     CX                      ; RECALL SCREEN BOUNDS
FF7C 52                         PUSH    DX                      ; RECALL [BH]=VISUAL PAGE
FF7D 33D2                       XOR     DX,DX                   ; WILL SET CURSOR POSITION TO [0,0]
                        ;-----------------------------------------------------------------
                        ;       THE LOOP FROM PRI10 TO THE INSTRUCTION PRIOR TO PRI20  :
                        ;       IS THE LOOP TO READ EACH CURSOR POSITION FROM THE      :
                        ;       SCREEN AND PRINT.                                      :
                        ;-----------------------------------------------------------------
FF7F                    PRI10:
FF7F B402                       MOV     AH,2                    ; TO INDICATE CURSOR SET REQUEST
FF81 CD10                       INT     10H                     ; NEW CURSOR POSITION ESTABLISHED
FF83 B408                       MOV     AH,8                    ; TO INDICATE READ CHARACTER
FF85 CD10                       INT     10H                     ; CHARACTER NOW IN [AL]
FF87 0AC0                       OR      AL,AL                   ; SEE IF VALID CHAR
FF89 7502                       JNZ     PRI15                   ; JUMP IF VALID CHAR
FF8B B020                       MOV     AL,' '                  ; MAKE A BLANK
FF8D                    PRI15:
FF8D 52                         PUSH    DX                      ; SAVE CURSOR POSITION
FF8E 33D2                       XOR     DX,DX                   ; INDICATE PRINTER 1
FF90 32E4                       XOR     AH,AH                   ; TO INDICATE PRINT CHAR IN [AL]
FF92 CD17                       INT     17H                     ; PRINT THE CHARACTER
FF94 5A                         POP     DX                      ; RECALL CURSOR POSITION
FF95 F6C425                     TEST    AH, 25H                 ; TEST FOR PRINTER ERROR
FF98 7521                       JNZ     ERR10                   ; JUMP IF ERROR DETECTED
FF9A FEC2                       INC     DL                      ; ADVANCE TO NEXT COLUMN
FF9C 3ACA                       CMP     CL,DL                   ; SEE IF AT END OF LINE
FF9E 75DF                       JNZ     PRI10                   ; IF NOT PROCEED
FFA0 32D2                       XOR     DL,DL                   ; BACK TO COLUMN 0
FFA2 8AE2                       MOV     AH,DL                   ; [AH]=0
FFA4 52                         PUSH    DX                      ; SAVE NEW CURSOR POSITION
FFA5 E82300                     CALL    CRLF                    ; LINE FEED CARRIAGE RETURN
FFA8 5A                         POP     DX                      ; RECALL CURSOR POSITION
FFA9 FEC6                       INC     DH                      ; ADVANCE TO NEXT LINE
FFAB 3AEE                       CMP     CH,DH                   ; FINISHED?
FFAD 75D0                       JNZ     PRI10                   ; IF NOT CONTINUE
FFAF                    PRI20:
FFAF 5A                         POP     DX                      ; RECALL CURSOR POSITION
FFB0 B402                       MOV     AH,2                    ; TO INDICATE CURSOR SET REQUEST
FFB2 CD10                       INT     10H                     ; CURSOR POSITION RESTORED
FFB4 C606000000                 MOV     STATUS_BYTE,0           ; INDICATE FINISHED
FFB9 EB0A                       JMP     SHORT EXIT              ; EXIT THE ROUTINE
FFBB                    ERR10:
```

Appendix A

```
LOC OBJ              LINE    SOURCE

FFBB 5A                               POP     DX                      ; GET CURSOR POSITION
FFBC B402                             MOV     AH,2                    ; TO REQUEST CURSOR SET
FFBE CD10                             INT     10H                     ; CURSOR POSITION RESTORED
FFC0                          ERR20:
FFC0 C6060000FF                       MOV     STATUS_BYTE,0FFH        ; INDICATE ERROR
FFC5                          EXIT:
FFC5 5A                               POP     DX                      ; RESTORE ALL THE REGISTERS USED
FFC6 59                               POP     CX
FFC7 5B                               POP     BX
FFC8 58                               POP     AX
FFC9 1F                               POP     DS
FFCA CF                               IRET
                              PRINT_SCREEN    ENDP

                              ;------ CARRIAGE RETURN, LINE FEED SUBROUTINE

FFCB                          CRLF    PROC    NEAR
FFCB 33D2                             XOR     DX,DX                   ; PRINTER 0
FFCD 32E4                             XOR     AH,AH                   ; WILL NOW SEND INITIAL LF,CR
                                                                      ;  TO PRINTER
FFCF B00A                             MOV     AL,12Q                  ; LF
FFD1 CD17                             INT     17H                     ; SEND THE LINE FEED
FFD3 32E4                             XOR     AH,AH                   ; NOW FOR THE CR
FFD5 B00D                             MOV     AL,15Q                  ; CR
FFD7 CD17                             INT     17H                     ; SEND THE CARRIAGE RETURN
FFD9 C3                               RET
                              CRLF    ENDP

FFDA 5041524954592043         D1      DB      'PARITY CHECK 2',13,10
     434845434B2032
FFE8 0D
FFE9 0A
FFEA 363031                   F3      DB      '601',13,10
FFED 0D
FFEE 0A

----                          CODE    ENDS

                              ;--------------------------------
                              ;       POWER ON RESET VECTOR    :
                              ;--------------------------------
----                          VECTOR  SEGMENT AT 0FFFFH

                              ;----- POWER ON RESET

0000 EA5BE000F0                       JMP     RESET

0005 31302F32372F3832                 DB      '10/27/82'              ; RELEASE MARKER
     32
----                          VECTOR  ENDS
                                      END
```

```
LOC OBJ          LINE     SOURCE

$TITLE(FIXEO OISK BIOS FOR IBH OISK CONTROLLER)

;-- INT 13 -------------------------------------------------------
;                                                                :
; FIXEO OISK I/O INTERFACE                                       :
;                                                                :
;       THIS INTERFACE PROVIDES ACCESS TO S 1/4" FIXED DISKS     :
;       THROUGH THE IBH FIXEO OISK CONTROLLER.                   :
;                                                                :
;-----------------------------------------------------------------

;-----------------------------------------------------------------
;       THE  BIOS ROUTINES ARE HEANT TO BE ACCESSEO THROUGH      :
;       SOFTWARE INTERRUPTS ONLY.  AHY ADDRESSES PRESENT IN      :
;       THE LISTINGS  ARE INCLUDEO  ONLY FOR  COMPLETENESS,      :
;       NOT FOR  REFERENCE.  APPLICATIONS WHICH  REFERENCE       :
;       ABSOLUTE  AODRESSES  WITHIN  THE  CODE  SEGMENT          :
;       VIOLATE THE STRUCTURE AND OESIGN OF BIOS.                :
;-----------------------------------------------------------------
;
; INPUT    (AH = HEX VALUE)
;
;       (AH)=00 RESET OISK (OL = 80H,81H) / OISKETTE
;       (AH)=01 READ THE STATUS OF THE LAST OISK OPERATION INTO (AL)
;               NOTE: OL < 80H - OISKETTE
;                     OL > 80H - OISK
;       (AH)=02 READ THE OESIREO SECTORS INTO HEHORY
;       (AH)=03 WRITE THE OESIREO SECTORS FROH HEHORY
;       (AH)=04 VERIFY THE OESIREO SECTORS
;       (AH)=05 FORMAT THE OESIRED TRACK
;       (AH)=06 FORMAT THE OESIREO TRACK AND SET BAD SECTOR FLAGS
;       (AH)=07 FORMAT THE ORIVE STARTING AT THE OESIREO TRACK
;       (AH)=08 RETURN THE CURRENT ORIVE PARAMETERS
;
;       (AH)=09 INITIALIZE ORIVE PAIR CHARACTERISTICS
;               INTERRUPT 41 POINTS TO OATA BLOCK
;       (AH)=0A READ LONG
;       (AH)=0B WRITE LONG
;       NOTE: READ AND WRITE LONG ENCOMPASS 512 + 4 BYTES ECC
;       (AH)=0C SEEK
;       (AH)=0D ALTERNATE OISK RESET (SEE OL)
;       (AH)=0E READ SECTOR BUFFER
;       (AH)=0F WRITE SECTOR BUFFER,
;               (RECOMMENDEO PRACTICE BEFORE FORMATTING)
;       (AH)=10 TEST ORIVE READY
;       (AH)=11 RECALIBRATE
;       (AH)=12 CONTROLLER RAM OIAGNOSTIC
;       (AH)=13 ORIVE OIAGNOSTIC
;       (AH)=14 CONTROLLER INTERNAL OIAGNOSTIC
;
;               REGISTERS USEO FOR FIXED OISK OPERATIONS
;
;               (OL)    - ORIVE NUMBER     (80H-87H FOR OISK, VALUE CHECKEO)
;               (OH)    - HEAD NUMBER      (0-7 ALLOWED, NOT VALUE CHECKEO)
;               (CH)    - CYLINDER NUMBER  (0-1023, NOT VALUE CHECKED)(SEE CL)
;               (CL)    - SECTOR NUMBER    (1-17, NOT VALUE CHECKEO)
;
;                         NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACEO
;                               IN THE HIGH 2 BITS OF THE CL REGISTER
;                               (10 BITS TOTAL)
;               (AL)    - NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,
;                                         FOR READ/WRITE LONG 1-79H)
;                         (INTERLEAVE VALUE FOR FORHAT 1-16D)
;               (ES:BX) - AODRESS OF BUFFER FOR READS AND WRITES,
;                         (NOT REQUIRED FOR VERIFY)
;
; OUTPUT
;       AH = STATUS OF CURRENT OPERATION
;            STATUS BITS ARE DEFINEO IN THE EQUATES BELOW
;       CY = 0  SUCCESSFUL OPERATION (AH=0 ON RETURN)
;       CY = 1  FAILEO OPERATION (AH HAS ERROR REASON)
;
;       NOTE:   ERROR 11H  INDICATES THAT THE OATA READ HAD A RECOVERABLE
;               ERROR WHICH WAS CORRECTEO BY THE ECC ALGORITHH.  THE OATA
;               IS PROBABLY GOOD,  HOWEVER THE BIOS ROUTINE INDICATES AN
;               ERROR TO ALLOW THE CONTROLLING PROGRAH A CHANCE TO DECIDE
;               FOR  ITSELF.  THE  ERROR  HAY  NOT  RECUR  IF THE DATA IS
```

Appendix A

```
LOC OBJ        LINE   SOURCE

                      ;               REWRITTEN. (AL) CONTAINS THE BURST LENGTH.
                      ;
                      ;       IF DRIVE PARAMETERS WERE REQUESTED,
                      ;
                      ;       DL = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (0-2)
                      ;               (CONTROLLER CARD ZERO TALLY ONLY)
                      ;       DH = MAXIMUM USEABLE VALUE FOR HEAD NUMBER
                      ;       CH = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER
                      ;       CL = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER
                      ;            AND CYLINDER NUMBER HIGH BITS
                      ;
                      ;       REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN
                      ;       INFORMATION.
                      ;
                      ;       NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE
                      ;               ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.
                      ;
                      ;-----------------------------------------------------------------------

 00FF                 SENSE_FAIL      EQU     0FFH            ; SENSE OPERATION FAILED
 00BB                 UNDEF_ERR       EQU     0BBH            ; UNDEFINED ERROR OCCURRED
 0080                 TIME_OUT        EQU     80H             ; ATTACHMENT FAILED TO RESPOND
 0040                 BAD_SEEK        EQU     40H             ; SEEK OPERATION FAILED
 0020                 BAD_CNTLR       EQU     20H             ; CONTROLLER HAS FAILED
 0011                 DATA_CORRECTED  EQU     11H             ; ECC CORRECTED DATA ERROR
 0010                 BAD_ECC         EQU     10H             ; BAD ECC ON DISK READ
 000B                 BAD_TRACK       EQU     0BH             ; BAD TRACK FLAG DETECTED
 0009                 DMA_BOUNDARY    EQU     09H             ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
 0007                 INIT_FAIL       EQU     07H             ; DRIVE PARAMETER ACTIVITY FAILED
 0005                 BAD_RESET       EQU     05H             ; RESET FAILED
 0004                 RECORD_NOT_FND  EQU     04H             ; REQUESTED SECTOR NOT FOUND
 0002                 BAD_ADDR_MARK   EQU     02H             ; ADDRESS MARK NOT FOUND
 0001                 BAD_CMD         EQU     01H             ; BAD COMMAND PASSED TO DISK I/O

                      ;----------------------------------------
                      ;       INTERRUPT AND STATUS AREAS      :
                      ;----------------------------------------

----                  DUMMY   SEGMENT AT 0
0034                          ORG     0DH*4                   ; FIXED DISK INTERRUPT VECTOR
0034                  HDISK_INT       LABEL   DWORD
004C                          ORG     13H*4                   ; DISK INTERRUPT VECTOR
004C                  ORG_VECTOR      LABEL   DWORD
0064                          ORG     19H*4                   ; BOOTSTRAP INTERRUPT VECTOR
0064                  BOOT_VEC        LABEL   DWORD
0078                          ORG     1EH*4                   ; DISKETTE PARAMETERS
0078                  DISKETTE_PARM   LABEL   DWORD
0100                          ORG     040H*4                  ; NEW DISKETTE INTERRUPT VECTOR
0100                  DISK_VECTOR     LABEL   DWORD
0104                          ORG     041H*4                  ; FIXED DISK PARAMETER VECTOR
0104                  HF_TBL_VEC      LABEL   DWORD
7C00                          ORG     7C00H                   ; BOOTSTRAP LOADER VECTOR
7C00                  BOOT_LOCN       LABEL   FAR
----                  DUMMY   ENDS

----                  DATA    SEGMENT AT 40H
0042                          ORG     42H
0042                  CMD_BLOCK       LABEL   BYTE
0042 (7 ??)           HD_ERROR        DB      7 DUP(?)        ; OVERLAYS DISKETTE STATUS
006C                          ORG     06CH
006C ????             TIMER_LOW       DW      ?               ; TIMER LOW WORD
0072                          ORG     72H
0072 ????             RESET_FLAG      DW      ?               ; 1234H IF KEYBOARD RESET UNDERWAY
0074                          ORG     74H
0074 ??               DISK_STATUS     DB      ?               ; FIXED DISK STATUS BYTE
0075 ??               HF_NUM          DB      ?               ; COUNT OF FIXED DISK DRIVES
0076 ??               CONTROL_BYTE    DB      ?               ; CONTROL BYTE DRIVE OPTIONS
0077 ??               PORT_OFF        DB      ?               ; PORT OFFSET
----                  DATA    ENDS

----                  CODE    SEGMENT

                      ;-------------------------------------------------------
                      ; HARDWARE SPECIFIC VALUES                              :
                      ;                                                       :
                      ;  -  CONTROLLER I/O PORT                               :
                      ;       > WHEN READ FROM:                               :
```

```
LOC OBJ                    SOURCE

                           ;        HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU)   :
                           ;        HF_PORT+1 - READ CONTROLLER HARDWARE STATUS      :
                           ;                    (CONTROLLER TO CPU)                  :
                           ;        HF_PORT+2 - READ CONFIGURATION SWITCHES          :
                           ;        HF_PORT+3 - NOT USED                             :
                           ;      > WHEN WRITTEN TO:                                 :
                           ;        HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER)  :
                           ;        HF_PORT+1 - CONTROLLER RESET                     :
                           ;        HF_PORT+2 - GENERATE CONTROLLER SELECT PULSE     :
                           ;        HF_PORT+3 - WRITE PATTERN TO DMA AND INTERRUPT   :
                           ;                    MASK REGISTER                        :
                           ;                                                         :
                           ;----------------------------------------------------------

 0320                      HF_PORT         EQU     0320H           ; DISK PORT
 0008                      R1_BUSY         EQU     00001000B       ; DISK PORT 1 BUSY BIT
 0004                      R1_BUS          EQU     00000100B       ;               COMMAND/DATA BIT
 0002                      R1_IOMODE       EQU     00000010B       ;               MODE BIT
 0001                      R1_REQ          EQU     00000001B       ;               REQUEST BIT

 0047                      DMA_READ        EQU     01000111B       ; CHANNEL 3 (047H)
 004B                      DMA_WRITE       EQU     01001011B       ; CHANNEL 3 (04BH)
 0000                      DMA             EQU     0               ; DMA ADDRESS
 0082                      DMA_HIGH        EQU     082H            ; PORT FOR HIGH 4 BITS OF DMA

 0000                      TST_RDY_CMD     EQU     00000000B       ; CNTLR READY (00H)
 0001                      RECAL_CMD       EQU     00000001B       ;       RECAL (01H)
 0003                      SENSE_CMD       EQU     00000011B       ;       SENSE (03H)
 0004                      FMTDRV_CMD      EQU     00000100B       ;       DRIVE (04H)
 0005                      CHK_TRK_CMD     EQU     00000101B       ;       T CHK (05H)
 0006                      FMTTRK_CMD      EQU     00000110B       ;       TRACK (06H)
 0007                      FMTBAD_CMD      EQU     00000111B       ;       BAD   (07H)
 0008                      READ_CMD        EQU     00001000B       ;       READ  (08H)
 000A                      WRITE_CMD       EQU     00001010B       ;       WRITE (0AH)
 000B                      SEEK_CMD        EQU     00001011B       ;       SEEK  (0BH)
 000C                      INIT_DRV_CMD    EQU     00001100B       ;       INIT  (0CH)
 000D                      RD_ECC_CMD      EQU     00001101B       ;       BURST (0DH)
 000E                      RD_BUFF_CMD     EQU     00001110B       ;       BUFFR (0EH)
 000F                      WR_BUFF_CMD     EQU     00001111B       ;       BUFFR (0FH)
 00E0                      RAM_DIAG_CMD    EQU     11100000B       ;       RAM   (E0H)
 00E3                      CHK_DRV_CMD     EQU     11100011B       ;       DRV   (E3H)
 00E4                      CNTLR_DIAG_CMD  EQU     11100100B       ;       CNTLR (E4H)
 00E5                      RD_LONG_CMD     EQU     11100101B       ;       RLONG (E5H)
 00E6                      WR_LONG_CMD     EQU     11100110B       ;       WLONG (E6H)

 0020                      INT_CTL_PORT    EQU     20H             ; 8259 CONTROL PORT
 0020                      EOI             EQU     20H             ; END OF INTERRUPT COMMAND

 0008                      MAX_FILE        EQU     8
 0002                      S_MAX_FILE      EQU     2

                                   ASSUME  CS:CODE
0000                               ORG     0H
0000 55                            DB      055H                    ; GENERIC BIOS HEADER
0001 AA                            DB      0AAH
0002 10                            DB      16D

                           ;----------------------------------------------------------------
                           ; FIXED DISK I/O SETUP                                          :
                           ;                                                               :
                           ;  -  ESTABLISH TRANSFER VECTORS FOR THE FIXED DISK             :
                           ;  -  PERFORM POWER ON DIAGNOSTICS                              :
                           ;     SHOULD AN ERROR OCCUR A "1701" MESSAGE IS DISPLAYED       :
                           ;                                                               :
                           ;----------------------------------------------------------------

0003                       DISK_SETUP      PROC    FAR
0003 EB1E                          JMP     SHORT   L3
0005 35303030303539                DB      '5000059 (C)COPYRIGHT  IBM 1982'        ; COPYRIGHT NOTICE
     20284329434F50
     59524947485420
     2049424D203139
     3832
0023                       L3:
                                   ASSUME  DS:DUMMY
0023 2BC0                          SUB     AX,AX                           ; ZERO
0025 8ED8                          MOV     DS,AX
```

Appendix A

```
LOC OBJ                 SOURCE

0027 FA                         CLI
0028 A14C00                     MOV     AX,WORD PTR ORG_VECTOR                 ; GET DISKETTE VECTOR
002B A30001                     MOV     WORD PTR DISK_VECTOR,AX                ;  INTO INT 40H
002E A14E00                     MOV     AX,WORD PTR ORG_VECTOR+2
0031 A30201                     MOV     WORD PTR DISK_VECTOR+2,AX
0034 C7064C005602               MOV     WORD PTR ORG_VECTOR, OFFSET DISK_IO    ; HDISK HANDLER
003A 8C0E4E00                   MOV     WORD PTR ORG_VECTOR+2,CS
003E B86007                     MOV     AX, OFFSET HD_INT                      ; HDISK INTERRUPT
0041 A33400                     MOV     WORD PTR HDISK_INT,AX
0044 8C0E3600                   MOV     WORD PTR HDISK_INT+2,CS
0048 C70664008601               MOV     WORD PTR BOOT_VEC,OFFSET BOOT_STRAP    ; BOOTSTRAP
004E 8C0E6600                   MOV     WORD PTR BOOT_VEC+2,CS
0052 C7060401E703               MOV     WORD PTR HF_TBL_VEC,OFFSET FD_TBL      ; PARAMETER TBL
0058 8C0E0601                   MOV     WORD PTR HF_TBL_VEC+2,CS
005C FB                         STI

                                ASSUME  DS:DATA
005D B84000                     MOV     AX,DATA                     ; ESTABLISH SEGMENT
0060 8ED8                       MOV     DS,AX
0062 C606740000                 MOV     DISK_STATUS,0               ; RESET THE STATUS INDICATOR
0067 C606750000                 MOV     HF_NUM,0                    ; ZERO COUNT OF DRIVES
006C C606430000                 MOV     CMD_BLOCK+1,0               ; DRIVE ZERO, SET VALUE IN BLOCK
0071 C606770000                 MOV     PORT_OFF,0                  ; ZERO CARD OFFSET

0076 B92500                     MOV     CX,25H                      ; RETRY COUNT
0079                    L4:
0079 E8F200                     CALL    HD_RESET_1                  ; RESET CONTROLLER
007C 7305                       JNC     L7
007E E2F9                       LOOP    L4                          ; TRY RESET AGAIN
0080 E9BF00                     JMP     ERROR_EX
0083                    L7:
0083 B90100                     MOV     CX,1
0086 BA8000                     MOV     DX,80H

0089 B80012                     MOV     AX,1200H                    ; CONTROLLER DIAGNOSTICS
008C CD13                       INT     13H
008E 7303                       JNC     P7
0090 E9AF00                     JMP     ERROR_EX
0093                    P7:
0093 B80014                     MOV     AX,1400H                    ; CONTROLLER DIAGNOSTICS
0096 CD13                       INT     13H
0098 7303                       JNC     P9
009A E9A500                     JMP     ERROR_EX
009D                    P9:
009D C7066C000000               MOV     TIMER_LOW,0                 ; ZERO TIMER
00A3 A17200                     MOV     AX,RESET_FLAG
00A6 3D3412                     CMP     AX,1234H                    ; KEYBOARD RESET
00A9 7506                       JNE     P8
00AB C7066C009A01               MOV     TIMER_LOW,410D              ; SKIP WAIT ON RESET
00B1                    P8:
00B1 E421                       IN      AL,021H                     ; TIMER
00B3 24FE                       AND     AL,0FEH                     ; ENABLE TIMER
00B5 E621                       OUT     021H,AL                     ; START TIMER
00B7                    P4:
00B7 E8B400                     CALL    HD_RESET_1                  ; RESET CONTROLLER
00BA 7207                       JC      P10
00BC B80010                     MOV     AX,1000H                    ; READY
00BF CD13                       INT     13H
00C1 730B                       JNC     P2
00C3                    P10:
00C3 A16C00                     MOV     AX,TIMER_LOW
00C6 3DBE01                     CMP     AX,446D                     ; 25 SECONDS
00C9 72EC                       JB      P4
00CB EB7590                     JMP     ERROR_EX
00CE                    P2:
00CE B90100                     MOV     CX,1
00D1 BA8000                     MOV     DX,80H

00D4 B80011                     MOV     AX,1100H                    ; RECALIBRATE
00D7 CD13                       INT     13H
00D9 7267                       JC      ERROR_EX

00DB B80009                     MOV     AX,0900H                    ; SET DRIVE PARAMETERS
00DE CD13                       INT     13H
00E0 7260                       JC      ERROR_EX

00E2 B800C8                     MOV     AX,0C800H                   ; DMA TO BUFFER
```

```
LOC OBJ                SOURCE

00E5 8EC0                      MOV     ES,AX                   ; SET SEGMENT
00E7 2BDB                      SUB     BX,BX
00E9 B8000F                    MOV     AX,0F00H                ; WRITE SECTOR BUFFER
00EC CD13                      INT     13H
00EE 7252                      JC      ERROR_EX

00F0 FE067500                  INC     HF_NUM                  ; DRIVE ZERO RESPONDED

00F4 BA1302                    MOV     DX,213H                 ; EXPANSION BOX
00F7 B000                      MOV     AL,0
00F9 EE                        OUT     DX,AL                   ; TURN BOX OFF
00FA BA2103                    MOV     DX,321H                 ; TEST IF CONTROLLER
00FD EC                        IN      AL,DX                   ; ... IS IN THE SYSTEM UNIT
00FE 240F                      AND     AL,0FH
0100 3C0F                      CMP     AL,0FH
0102 7406                      JE      BOX_ON
0104 C7066C00A401              MOV     TIMER_LOW,420D          ; CONTROLLER IS IN SYSTEM UNIT
010A                   BOX_ON:
010A BA1302                    MOV     DX,213H                 ; EXPANSION BOX
010D B0FF                      MOV     AL,0FFH
010F EE                        OUT     DX,AL                   ; TURN BOX ON

0110 B90100                    MOV     CX,1                    ; ATTEMPT NEXT DRIVES
0113 BA8100                    MOV     DX,081H
0116                   P3:
0116 2BC0                      SUB     AX,AX                   ; RESET
0118 CD13                      INT     13H
011A 7240                      JC      POD_DONE
011C B80011                    MOV     AX,01100H               ; RECAL
011F CD13                      INT     13H
0121 730B                      JNC     P5
0123 A16C00                    MOV     AX,TIMER_LOW
0126 3DBE01                    CMP     AX,446D                 ; 25 SECONDS
0129 72EB                      JB      P3
012B EB2F90                    JMP     POD_DONE
012E                   P5:
012E B80009                    MOV     AX,0900H                ; INITIALIZE CHARACTERISTICS
0131 CD13                      INT     13H
0133 7227                      JC      POD_DONE
0135 FE067500                  INC     HF_NUM                  ; TALLY ANOTHER DRIVE
0139 81FA8100                  CMP     DX,(80H + S_MAX_FILE - 1)
013D 731D                      JAE     POD_DONE
013F 42                        INC     DX
0140 EBD4                      JMP     P3

                       ;----- POD ERROR

0142                   ERROR_EX:
0142 BD0F00                    MOV     BP,0FH                  ; POD ERROR FLAG
0145 2BC0                      SUB     AX,AX
0147 8BF0                      MOV     SI,AX
0149 B9060090                  MOV     CX,F17L                 ; MESSAGE CHARACTER COUNT
014D B700                      MOV     BH,0                    ; PAGE ZERO
014F                   OUT_CH:
014F 2E8A846801                MOV     AL,CS:F17[SI]           ; GET BYTE
0154 B40E                      MOV     AH,14D                  ; VIDEO OUT
0156 CD10                      INT     10H                     ; DISPLAY CHARACTER
0158 46                        INC     SI                      ; NEXT CHAR
0159 E2F4                      LOOP    OUT_CH                  ; DO MORE
015B F9                        STC
015C                   POD_DONE:
015C FA                        CLI
015D E421                      IN      AL,021H                 ; BE SURE TIMER IS DISABLED
015F 0C01                      OR      AL,01H
0161 E621                      OUT     021H,AL
0163 FB                        STI
0164 E8A500                    CALL    DSBL
0167 CB                        RET

0168 31373031          F17     DB      '1701',0DH,0AH
```


Appendix A

```
LOC OBJ          SOURCE

016C 0D
016D 0A
  0006                  F17L    EQU     $-F17

016E                    HD_RESET_1      PROC    NEAR
016E 51                         PUSH    CX                      ; SAVE REGISTER
016F 52                         PUSH    DX
0170 F8                         CLC                             ; CLEAR CARRY
0171 B90001                     MOV     CX,0100H                ; RETRY COUNT
0174                    L6:
0174 E80706                     CALL    PORT_1
0177 EE                         OUT     DX,AL                   ; RESET CARD
0178 E80306                     CALL    PORT_1
017B EC                         IN      AL,DX                   ; CHECK STATUS
017C 2402                       AND     AL,2                    ; ERROR BIT
017E 7403                       JZ      R3
0180 E2F2                       LOOP    L6
0182 F9                         STC
0183                    R3:
0183 5A                         POP     DX                      ; RESTORE REGISTER
0184 59                         POP     CX
0185 C3                         RET
                        HD_RESET_1      ENDP

                        DISK_SETUP      ENDP

                        ;----- INT 19 -----------------------------------------------
                        ;                                                          :
                        ; INTERRUPT 19 BOOT STRAP LOADER                           :
                        ;                                                          :
                        ;  - THE FIXED DISK BIOS REPLACES THE INTERRUPT 19         :
                        ;    BOOT STRAP VECTOR WITH A POINTER TO THIS BOOT ROUTINE :
                        ;  - RESET THE DEFAULT DISK AND DISKETTE PARAMETER VECTORS :
                        ;  - THE BOOT BLOCK TO BE READ IN WILL BE ATTEMPTED FROM   :
                        ;    CYLINDER 0 SECTOR 1 OF THE DEVICE.                    :
                        ;  - THE BOOTSTRAP SEQUENCE IS:                            :
                        ;    > ATTEMPT TO LOAD FROM THE DISKETTE INTO THE BOOT     :
                        ;      LOCATION (0000:7C00) AND TRANSFER CONTROL THERE     :
                        ;    > IF THE DISKETTE FAILS THE FIXED DISK IS TRIED FOR A :
                        ;      VALID BOOTSTRAP BLOCK. A VALID BOOT BLOCK ON THE    :
                        ;      FIXED DISK CONSISTS OF THE BYTES 055H 0AAH AS THE   :
                        ;      LAST TWO BYTES OF THE BLOCK                         :
                        ;    > IF THE ABOVE FAILS CONTROL IS PASSED TO RESIDENT BASIC :
                        ;                                                          :
                        ;-----------------------------------------------------------

01B6                    BOOT_STRAP:
                                ASSUME  DS:DUMMY,ES:DUMMY
01B6 2BC0                       SUB     AX,AX
01B8 8ED8                       MOV     DS,AX                   ; ESTABLISH SEGMENT

                        ;----- RESET PARAMETER VECTORS

01BA FA                         CLI
01BB C7060401E703               MOV     WORD PTR HF_TBL_VEC, OFFSET FD_TBL
0191 8C0E0601                   MOV     WORD PTR HF_TBL_VEC+2, CS
0195 C70678000102               MOV     WORD PTR DISKETTE_PARM, OFFSET DISKETTE_TBL
019B 8C0E7A00                   MOV     WORD PTR DISKETTE_PARM+2, CS
019F FB                         STI

                        ;----- ATTEMPT BOOTSTRAP FROM DISKETTE

01A0 B90300                     MOV     CX,3                    ; SET RETRY COUNT
01A3                    H1:                                     ; IPL_SYSTEM
01A3 51                         PUSH    CX                      ; SAVE RETRY COUNT
01A4 2BD2                       SUB     DX,DX                   ; DRIVE ZERO
01A6 2BC0                       SUB     AX,AX                   ; RESET THE DISKETTE
01A8 CD13                       INT     13H                     ; FILE IO CALL
01AA 720F                       JC      H2                      ; IF ERROR, TRY AGAIN
01AC B80102                     MOV     AX,0201H                ; READ IN THE SINGLE SECTOR

01AF 2BD2                       SUB     DX,DX
01B1 8EC2                       MOV     ES,DX                   ; ESTABLISH SEGMENT
01B3 BB007C                     MOV     BX,OFFSET BOOT_LOCN

01B6 B90100                     MOV     CX,1                    ; SECTOR 1, TRACK 0
01B9 CD13                       INT     13H                     ; FILE IO CALL
```

```
LOC OBJ            SOURCE

01BB 59            M2:     POP     CX                      ; RECOVER RETRY COUNT
01BC 730A                  JNC     M4                      ; CF SET BY UNSUCCESSFUL READ
01BE 80FC80                CMP     AH,80H                  ; IF TIME OUT, NO RETRY
01C1 740A                  JZ      M5                      ; TRY FIXED DISK
01C3 E2DE                  LOOP    M1                      ; DO IT FOR RETRY TIMES
01C5 EB0690                JMP     M5                      ; UNABLE TO IPL FROM THE DISKETTE
01C8               M4:                                     ; IPL WAS SUCCESSFUL
01C8 EA007C0000            JMP     BOOT_LOCN

                   ;----- ATTEMPT BOOTSTRAP FROM FIXED DISK

01CD               M5:
01CD 2BC0                  SUB     AX,AX                   ; RESET DISKETTE
01CF 2BD2                  SUB     DX,DX
01D1 CD13                  INT     13H
01D3 B90300                MOV     CX,3                    ; SET RETRY COUNT
01D6               M6:                                     ; IPL_SYSTEM
01D6 51                    PUSH    CX                      ; SAVE RETRY COUNT
01D7 BA8000                MOV     DX,0080H                ; FIXED DISK ZERO
01DA 2BC0                  SUB     AX,AX                   ; RESET THE FIXED DISK
01DC CD13                  INT     13H                     ; FILE IO CALL
01DE 7212                  JC      M7                      ; IF ERROR, TRY AGAIN
01E0 B80102                MOV     AX,0201H                ; READ IN THE SINGLE SECTOR
01E3 2BDB                  SUB     BX,BX
01E5 8EC3                  MOV     ES,BX
01E7 BB007C                MOV     BX,OFFSET BOOT_LOCN     ; TO THE BOOT LOCATION
01EA BA8000                MOV     DX,80H                  ; DRIVE NUMBER
01ED B90100                MOV     CX,1                    ; SECTOR 1, TRACK 0
01F0 CD13                  INT     13H                     ; FILE IO CALL
01F2 59            M7:     POP     CX                      ; RECOVER RETRY COUNT
01F3 7208                  JC      M8
01F5 A1FE7D                MOV     AX,WORD PTR BOOT_LOCN+510
01F8 3D55AA                CMP     AX,0AA55H               ; TEST FOR GENERIC BOOT BLOCK
01FB 74CB                  JZ      M4
01FD               M8:
01FD E2D7                  LOOP    M6                      ; DO IT FOR RETRY TIMES

                   ;------ UNABLE TO IPL FROM THE DISKETTE OR FIXED DISK

01FF CD18                  INT     18H                     ; RESIDENT BASIC

0201               DISKETTE_TBL:

0201 CF                    DB      11001111B               ; SRT=C, HD UNLOAD=0F - 1ST SPEC BYTE
0202 02                    DB      2                       ; HD LOAD=1, MODE=DMA - 2ND SPEC BYTE
0203 25                    DB      25H                     ; WAIT AFTER OPN TIL MOTOR OFF
0204 02                    DB      2                       ; 512 BYTES PER SECTOR
0205 08                    DB      8                       ; EOT (LAST SECTOR ON TRACK)
0206 2A                    DB      02AH                    ; GAP LENGTH
0207 FF                    DB      0FFH                    ; DTL
0208 50                    DB      050H                    ; GAP LENGTH FOR FORMAT
0209 F6                    DB      0F6H                    ; FILL BYTE FOR FORMAT
020A 19                    DB      25                      ; HEAD SETTLE TIME (MILLISECONDS)
020B 04                    DB      4                       ; MOTOR START TIME (1/8 SECOND)

                   ;----- MAKE SURE THAT ALL HOUSEKEEPING IS DONE BEFORE EXIT

020C               DSBL    PROC    NEAR
                           ASSUME  DS:DATA
020C 1E                    PUSH    DS                      ; SAVE SEGMENT
020D B84000                MOV     AX,DATA
0210 8ED8                  MOV     DS,AX

0212 8A267700              MOV     AH,PORT_OFF
0216 50                    PUSH    AX                      ; SAVE OFFSET

0217 C606770000            MOV     PORT_OFF,0H
021C E86905                CALL    PORT_3
021F 2AC0                  SUB     AL,AL
0221 EE                    OUT     DX,AL                   ; RESET INT/DMA MASK
0222 C606770004            MOV     PORT_OFF,4H
0227 E85E05                CALL    PORT_3
022A 2AC0                  SUB     AL,AL
022C EE                    OUT     DX,AL                   ; RESET INT/DMA MASK
022D C606770008            MOV     PORT_OFF,8H
0232 E85305                CALL    PORT_3
0235 2AC0                  SUB     AL,AL
```

Appendix A

```
LOC OBJ                LINE    SOURCE

0237 EE                        OUT     DX,AL                   ; RESET INT/DMA MASK
0238 C60677000C                MOV     PORT_OFF,0CH
023D E84805                    CALL    PORT_3
0240 2AC0                      SUB     AL,AL
0242 EE                        OUT     DX,AL                   ; RESET INT/DMA MASK
0243 B007                      MOV     AL,07H
0245 E60A                      OUT     DMA+10,AL               ; SET DMA MODE TO DISABLE
0247 FA                        CLI                             ; DISABLE INTERRUPTS
0248 E421                      IN      AL,021H
024A 0C20                      OR      AL,020H
024C E621                      OUT     021H,AL                 ; DISABLE INTERRUPT 5
024E FB                        STI                             ; ENABLE INTERRUPTS
024F 58                        POP     AX                      ; RESTORE OFFSET
0250 88267700                  MOV     PORT_OFF,AH
0254 1F                        POP     DS                      ; RESTORE SEGMENT
0255 C3                        RET
                       DSBL    ENDP

                       ;----------------------------------------
                       ;       FIXED DISK BIOS ENTRY POINT      :
                       ;----------------------------------------

0256                   DISK_IO PROC    FAR
                               ASSUME  DS:NOTHING,ES:NOTHING
0256 80FA80                    CMP     DL,80H                  ; TEST FOR FIXED DISK DRIVE
0259 7305                      JAE     HARD_DISK               ; YES, HANDLE HERE
025B CD40                      INT     40H                     ; DISKETTE HANDLER
025D                   RET_2:
025D CA0200                    RET     2                       ; BACK TO CALLER
0260                   HARD_DISK:
                               ASSUME  DS:DATA
0260 FB                        STI                             ; ENABLE INTERRUPTS
0261 0AE4                      OR      AH,AH
0263 7509                      JNZ     A3
0265 CD40                      INT     40H                     ; RESET NEC WHEN AH=0
0267 2AE4                      SUB     AH,AH
0269 80FA81                    CMP     DL,(80H + S_MAX_FILE - 1)
026C 77EF                      JA      RET_2
026E                   A3:
026E 80FC08                    CMP     AH,08                   ; GET PARAMETERS IS A SPECIAL CASE
0271 7503                      JNZ     A2
0273 E91A01                    JMP     GET_PARM_N
0276                   A2:
0276 53                        PUSH    BX                      ; SAVE REGISTERS DURING OPERATION
0277 51                        PUSH    CX
0278 52                        PUSH    DX
0279 1E                        PUSH    DS
027A 06                        PUSH    ES
027B 56                        PUSH    SI
027C 57                        PUSH    DI

027D E86A00                    CALL    DISK_IO_CONT            ; PERFORM THE OPERATION

0280 50                        PUSH    AX
0281 E888FF                    CALL    DSBL                    ; BE SURE DISABLES OCCURRED
0284 B84000                    MOV     AX,DATA
0287 8ED8                      MOV     DS,AX                   ; ESTABLISH SEGMENT
0289 58                        POP     AX
028A 8A267400                  MOV     AH,DISK_STATUS          ; GET STATUS FROM OPERATION
028E 80FC01                    CMP     AH,1                    ; SET THE CARRY FLAG TO INDICATE
0291 F5                        CMC                             ;  SUCCESS OR FAILURE
0292 5F                        POP     DI                      ; RESTORE REGISTERS
0293 5E                        POP     SI
0294 07                        POP     ES
0295 1F                        POP     DS
0296 5A                        POP     DX
0297 59                        POP     CX
0298 5B                        POP     BX
0299 CA0200                    RET     2                       ; THROW AWAY SAVED FLAGS
                       DISK_IO ENDP

029C                   M1      LABEL   WORD                    ; FUNCTION TRANSFER TABLE
029C 3803                      DW      DISK_RESET              ; 000H
029E 4003                      DW      RETURN_STATUS           ; 001H
02A0 5603                      DW      DISK_READ               ; 002H
02A2 6003                      DW      DISK_WRITE              ; 003H
02A4 6A03                      DW      DISK_VERF               ; 004H
```

```
LOC OBJ                 SOURCE

02A6 7203                       DW      FMT_TRK                 ; 005H
02A8 7903                       DW      FMT_BAD                 ; 006H
02AA 8003                       DW      FMT_DRV                 ; 007H
02AC 3003                       DW      BAD_COMMAND             ; 008H
02AE 2704                       DW      INIT_DRV                ; 009H
02B0 CF04                       DW      RD_LONG                 ; 00AH
02B2 0D04                       DW      WR_LONG                 ; 00BH
02B4 F204                       DW      DISK_SEEK               ; 00CH
02B6 3803                       DW      DISK_RESET              ; 00DH
02B8 F904                       DW      RD_BUFF                 ; 00EH
02BA 0705                       DW      WR_BUFF                 ; 00FH
02BC 1505                       DW      TST_RDY                 ; 010H
02BE 1C05                       DW      HDISK_RECAL             ; 011H
02C0 2305                       DW      RAM_DIAG                ; 012H
02C2 2A05                       DW      CHK_DRV                 ; 013H
02C4 3105                       DW      CNTLR_DIAG              ; 014H
  002A                  M1L     EQU     $-M1

02C6                    SETUP_A PROC    NEAR

02C6 C606740000                 MOV     DISK_STATUS,0           ; RESET THE STATUS INDICATOR
02CB 51                         PUSH    CX                      ; SAVE CX

                        ;----- CALCULATE THE PORT OFFSET

02CC 8AEA                       MOV     CH,DL                   ; SAVE DL
02CE 80CA01                     OR      DL,1
02D1 FECA                       DEC     DL
02D3 D0E2                       SHL     DL,1                    ; GENERATE OFFSET
02D5 88167700                   MOV     PORT_OFF,DL             ; STORE OFFSET
02D9 8AD5                       MOV     DL,CH                   ; RESTORE DL
02DB 80E201                     AND     DL,1

02DE B105                       MOV     CL,5                    ; SHIFT COUNT
02E0 D2E2                       SHL     DL,CL                   ; DRIVE NUMBER (0,1)
02E2 0AD6                       OR      DL,DH                   ; HEAD NUMBER
02E4 88164300                   MOV     CMD_BLOCK+1,DL
02E8 59                         POP     CX
02E9 C3                         RET
                        SETUP_A ENDP

02EA                    DISK_IO_CONT    PROC    NEAR
02EA 50                         PUSH    AX
02EB B84000                     MOV     AX,DATA
02EE 8ED8                       MOV     DS,AX                   ; ESTABLISH SEGMENT
02F0 58                         POP     AX
02F1 80FC01                     CMP     AH,01H                  ; RETURN STATUS
02F4 7503                       JNZ     A4
02F6 EB5590                     JMP     RETURN_STATUS
02F9                    A4:
02F9 80EA80                     SUB     DL,80H                  ; CONVERT DRIVE NUMBER TO 0 BASED RANGE
02FC 80FA08                     CMP     DL,MAX_FILE             ; LEGAL DRIVE TEST
02FF 732F                       JAE     BAD_COMMAND

0301 E8C2FF                     CALL    SETUP_A

                        ;----- SET UP COMMAND BLOCK

0304 FEC9                       DEC     CL                      ; SECTORS 0-16 FOR CONTROLLER
0306 C606420000                 MOV     CMD_BLOCK+0,0
030B 880E4400                   MOV     CMD_BLOCK+2,CL          ; SECTOR AND HIGH 2 BITS CYLINDER
030F 882E4500                   MOV     CMD_BLOCK+3,CH          ; CYLINDER
0313 A24600                     MOV     CMD_BLOCK+4,AL          ; INTERLEAVE / BLOCK COUNT
0316 A07600                     MOV     AL,CONTROL_BYTE         ; CONTROL BYTE (STEP OPTION)
0319 A24700                     MOV     CMD_BLOCK+5,AL
031C 50                         PUSH    AX                      ; SAVE AX
031D 8AC4                       MOV     AL,AH                   ; GET INTO LOW BYTE
031F 32E4                       XOR     AH,AH                   ; ZERO HIGH BYTE
0321 D1E0                       SAL     AX,1                    ; *2 FOR TABLE LOOKUP
0323 8BF0                       MOV     SI,AX                   ; PUT INTO SI FOR BRANCH
0325 3D2A00                     CMP     AX,M1L                  ; TEST WITHIN RANGE
0328 58                         POP     AX                      ; RESTORE AX
0329 7305                       JNB     BAD_COMMAND
032B 2EFFA49C02                 JMP     WORD PTR CS:[SI + OFFSET M1]
0330                    BAD_COMMAND:
0330 C606740001                 MOV     DISK_STATUS,BAD_CMD     ; COMMAND ERROR
0335 B000                       MOV     AL,0
```

Appendix A

```
LOC OBJ                 SOURCE

0337 C3                         RET
                        DISK_IO_CONT    ENDP

                        ;-----------------------------------------------
                        ;       RESET THE DISK SYSTEM  (AH = 000H)     :
                        ;-----------------------------------------------

0338                    DISK_RESET      PROC    NEAR
0338 E84304                     CALL    PORT_1                  ; RESET PORT
033B EE                         OUT     DX,AL                   ; ISSUE RESET
033C E83F04                     CALL    PORT_1                  ; CONTROLLER HARDWARE STATUS
033F EC                         IN      AL,DX                   ; GET STATUS
0340 2402                       AND     AL,2                    ; ERROR BIT
0342 7406                       JZ      DR1
0344 C606740005                 MOV     DISK_STATUS,BAD_RESET
0349 C3                         RET
034A                    DR1:
034A E90A00                     JMP     INIT_DRV                ; SET THE DRIVE PARAMETERS
                        DISK_RESET      ENDP

                        ;-----------------------------------------------
                        ;       DISK STATUS ROUTINE   (AH = 001H)      :
                        ;-----------------------------------------------

034D                    RETURN_STATUS   PROC    NEAR
034D A07400                     MOV     AL,DISK_STATUS          ; OBTAIN PREVIOUS STATUS
0350 C606740000                 MOV     DISK_STATUS,0           ; RESET STATUS
0355 C3                         RET
                        RETURN_STATUS   ENDP

                        ;-----------------------------------------------
                        ;       DISK READ ROUTINE    (AH = 002H)       :
                        ;-----------------------------------------------

0356                    DISK_READ       PROC    NEAR
0356 B047                       MOV     AL,DMA_READ             ; MODE BYTE FOR DMA READ
0358 C60642000B                 MOV     CMD_BLOCK+0,READ_CMD
035D E9E501                     JMP     DMA_OPN
                        DISK_READ       ENDP

                        ;-----------------------------------------------
                        ;       DISK WRITE ROUTINE    (AH = 003H)      :
                        ;-----------------------------------------------

0360                    DISK_WRITE      PROC    NEAR
0360 B04B                       MOV     AL,DMA_WRITE            ; MODE BYTE FOR DMA WRITE
0362 C60642000A                 MOV     CMD_BLOCK+0,WRITE_CMD
0367 E9DB01                     JMP     DMA_OPN
                        DISK_WRITE      ENDP

                        ;-----------------------------------------------
                        ;       DISK VERIFY    (AH = 004H)             :
                        ;-----------------------------------------------

036A                    DISK_VERF       PROC    NEAR
036A C606420005                 MOV     CMD_BLOCK+0,CHK_TRK_CMD
036F E9C401                     JMP     NDMA_OPN
                        DISK_VERF       ENDP

                        ;-----------------------------------------------
                        ;       FORMATTING    (AH = 005H 006H 007H)    :
                        ;-----------------------------------------------

0372                    FMT_TRK PROC    NEAR                    ; FORMAT TRACK   (AH = 005H)
0372 C606420006                 MOV     CMD_BLOCK,FMTTRK_CMD
0377 EB0C                       JMP     SHORT   FMT_CONT
                        FMT_TRK ENDP

0379                    FMT_BAD PROC    NEAR                    ; FORMAT BAD TRACK  (AH = 006H)
0379 C606420007                 MOV     CMD_BLOCK,FMTBAD_CMD
037E EB05                       JMP     SHORT   FMT_CONT
                        FMT_BAD ENDP

0380                    FMT_DRV PROC    NEAR                    ; FORMAT DRIVE  (AH = 007H)
0380 C606420004                 MOV     CMD_BLOCK,FMTDRV_CMD
                        FMT_DRV ENDP
```

```
LOC OBJ              SOURCE

0385                 FMT_CONT:
0385 A04400                    MOV     AL,CMD_BLOCK+2          ; ZERO OUT SECTOR FIELD
0388 24C0                      AND     AL,11000000B
038A A24400                    MOV     CMD_BLOCK+2,AL
038D E9A601                    JMP     NDMA_OPN

                     ;-----------------------------------------------
                     ;         GET PARAMETERS   (AH = 8)             :
                     ;-----------------------------------------------

0390                 GET_PARM_N        LABEL   NEAR
0390                 GET_PARM          PROC    FAR             ; GET DRIVE PARAMETERS
0390 1E                        PUSH    DS                      ; SAVE REGISTERS
0391 06                        PUSH    ES
0392 53                        PUSH    BX

                               ASSUME  DS:DUMMY
0393 2BC0                      SUB     AX,AX                   ; ESTABLISH ADDRESSING
0395 8ED8                      MOV     DS,AX
0397 C41E0401                  LES     BX,HF_TBL_VEC
                               ASSUME  DS:DATA
039B B84000                    MOV     AX,DATA
039E 8ED8                      MOV     DS,AX                   ; ESTABLISH SEGMENT

03A0 80EA80                    SUB     DL,80H
03A3 80FA08                    CMP     DL,MAX_FILE             ; TEST WITHIN RANGE
03A6 732F                      JAE     G4

03A8 E81BFF                    CALL    SETUP_A

03AB E8DF03                    CALL    SW2_OFFS
03AE 7227                      JC      G4
03B0 03D8                      ADD     BX,AX

03B2 268B07                    MOV     AX,ES:[BX]              ; MAX NUMBER OF CYLINDERS
03B5 2D0200                    SUB     AX,2                    ; ADJUST FOR 0-N
                                                               ; AND RESERVE LAST TRACK
03B8 8AE8                      MOV     CH,AL
03BA 250003                    AND     AX,0300H                ; HIGH TWO BITS OF CYL
03BD D1E8                      SHR     AX,1
03BF D1E8                      SHR     AX,1
03C1 0C11                      OR      AL,011H                 ; SECTORS
03C3 8AC8                      MOV     CL,AL

03C5 268A7702                  MOV     DH,ES:[BX][2]           ; HEADS
03C9 FECE                      DEC     DH                      ; 0-N RANGE
03CB 8A167500                  MOV     DL,HF_NUM               ; DRIVE COUNT
03CF 2BC0                      SUB     AX,AX
03D1                 G5:
03D1 5B                        POP     BX                      ; RESTORE REGISTERS
03D2 07                        POP     ES
03D3 1F                        POP     DS
03D4 CA0200                    RET     2
03D7                 G4:
03D7 C606740007                MOV     DISK_STATUS,INIT_FAIL   ; OPERATION FAILED
03DC B407                      MOV     AH,INIT_FAIL
03DE 2AC0                      SUB     AL,AL
03E0 2BD2                      SUB     DX,DX
03E2 2BC9                      SUB     CX,CX
03E4 F9                        STC                             ; SET ERROR FLAG
03E5 EBEA                      JMP     G5
                     GET_PARM          ENDP

                     ;----------------------------------------------------------
                     ; INITIALIZE DRIVE CHARACTERISTICS                         :
                     ;                                                          :
                     ; FIXED DISK PARAMETER TABLE                               :
                     ;                                                          :
                     ;  -  THE TABLE IS COMPOSED OF A BLOCK DEFINED AS:         :
                     ;                                                          :
                     ;       (1 WORD) - MAXIMUM NUMBER OF CYLINDERS             :
                     ;       (1 BYTE) - MAXIMUM NUMBER OF HEADS                 :
                     ;       (1 WORD) - STARTING REDUCED WRITE CURRENT CYL      :
                     ;       (1 WORD) - STARTING WRITE PRECOMPENSATION CYL      :
                     ;       (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH           :
                     ;       (1 BYTE) - CONTROL BYTE (DRIVE STEP OPTION)        :
                     ;                  BIT    7 DISABLE DISK-ACCESS RETRIES    :
                     ;                  BIT    6 DISABLE ECC RETRIES            :
```

Appendix A

```
LOC OBJ                 SOURCE

                        ;               BITS 5-3 ZERO                           :
                        ;               BITS 2-0 DRIVE OPTION                   :
                        ;       (1 BYTE) - STANDARD TIME OUT VALUE (SEE BELOW)  :
                        ;       (1 BYTE) - TIME OUT VALUE FOR FORMAT DRIVE      :
                        ;       (1 BYTE) - TIME OUT VALUE FOR CHECK DRIVE       :
                        ;       (4 BYTES)                                       :
                        ;               - RESERVED FOR FUTURE USE               :
                        ;                                                       :
                        ;        - TO DYNAMICALLY DEFINE A SET OF PARAMETERS    :
                        ;          BUILD A TABLE OF VALUES AND PLACE THE        :
                        ;          CORRESPONDING VECTOR INTO INTERRUPT 41.      :
                        ;                                                       :
                        ;       NOTE:                                           :
                        ;               THE DEFAULT TABLE IS VECTORED IN FOR    :
                        ;               AN INTERRUPT 19H (BOOTSTRAP)            :
                        ;                                                       :
                        ;                                                       :
                        ; ON THE CARD SWITCH SETTINGS                           :
                        ;                                                       :
                        ;               DRIVE 0    DRIVE 1                      :
                        ;              ----------------------                   :
                        ;       ON     :           /        :                   :
                        ;              : -1-  -2- / -3-  -4- :                  :
                        ;       OFF    :           /        :                   :
                        ;              ----------------------                   :
                        ;                                                       :
                        ;                                                       :
                        ;       TRANSLATION TABLE                               :
                        ;                                                       :
                        ;       1/3  :  2/4  :  TABLE ENTRY                     :
                        ;       -------------------------                       :
                        ;        ON  :   ON  :      0                           :
                        ;        ON  :  OFF  :      1                           :
                        ;       OFF  :   ON  :      2                           :
                        ;       OFF  :  OFF  :      3                           :
                        ;                                                       :
                        ;--------------------------------------------------------

03E7                    FD_TBL:

                        ;----- DRIVE TYPE 00

03E7 3201                       DW      0306D
03E9 02                         DB      02D
03EA 3201                       DW      0306D
03EC 0000                       DW      0000D
03EE 0B                         DB      0BH
03EF 00                         DB      00H
03F0 0C                         DB      0CH                     ; STANDARD
03F1 B4                         DB      0B4H                    ; FORMAT DRIVE
03F2 28                         DB      028H                    ; CHECK DRIVE
03F3 00000000                   DB      0,0,0,0

                        ;----- DRIVE TYPE 01

03F7 7701                       DW      0375D
03F9 08                         DB      08D
03FA 7701                       DW      0375D
03FC 0000                       DW      0000D
03FE 0B                         DB      0BH
03FF 05                         DB      05H
0400 0C                         DB      0CH                     ; STANDARD
0401 B4                         DB      0B4H                    ; FORMAT DRIVE
0402 28                         DB      028H                    ; CHECK DRIVE
0403 00000000                   DB      0,0,0,0

                        ;----- DRIVE TYPE 02

0407 3201                       DW      0306D
0409 06                         DB      06D
040A 8000                       DW      0128D
040C 0001                       DW      0256D
040E 0B                         DB      0BH
040F 05                         DB      05H
0410 0C                         DB      0CH                     ; STANDARD
0411 B4                         DB      0B4H                    ; FORMAT DRIVE
```

```
LOC OBJ                 SOURCE

0412 28                           DB      028H                    ; CHECK DRIVE
0413 00000000                     DB      0,0,0,0
                        ;----- DRIVE TYPE 03
0417 3201                         DW      0306D
0419 04                           DB      04D
041A 3201                         DW      0306D
041C 0000                         DW      00000
041E 0B                           DB      0BH
041F 05                           DB      05H
0420 0C                           DB      0CH                     ; STANDARD
0421 B4                           DB      0B4H                    ; FORMAT DRIVE
0422 28                           DB      028H                    ; CHECK DRIVE
0423 00000000                     DB      0,0,0,0
0427                    INIT_DRV          PROC    NEAR
                        ;----- DO DRIVE ZERO
0427 C60642000C                   MOV     CMD_BLOCK+0,INIT_DRV_CMD
042C C606430000                   MOV     CMD_BLOCK+1,0
0431 E81000                       CALL    INIT_DRV_R
0434 720D                         JC      INIT_DRV_OUT
                        ;----- DO DRIVE ONE
0436 C60642000C                   MOV     CMD_BLOCK+0,INIT_DRV_CMD
043B C606430020                   MOV     CMD_BLOCK+1,00100000B
0440 E80100                       CALL    INIT_DRV_R
0443                    INIT_DRV_OUT:
0443 C3                           RET
                        INIT_DRV          ENDP
0444                    INIT_DRV_R        PROC    NEAR
                                  ASSUME  ES:CODE
0444 2AC0                         SUB     AL,AL
0446 E81901                       CALL    COMMAND                 ; ISSUE THE COMMAND
0449 7301                         JNC     B1
044B C3                           RET
044C                    B1:
044C 1E                           PUSH    DS                      ; SAVE SEGMENT
                                  ASSUME  DS:DUMMY
044D 2BC0                         SUB     AX,AX
044F 8ED8                         MOV     DS,AX                   ; ESTABLISH SEGMENT
0451 C41E0401                     LES     BX,HF_TBL_VEC
0455 1F                           POP     DS                      ; RESTORE SEGMENT
                                  ASSUME  DS:DATA
0456 E83403                       CALL    SW2_OFFS
0459 7257                         JC      B3
045B 03D8                         ADD     BX,AX
                        ;----- SEND DRIVE PARAMETERS MOST SIGNIFICANT BYTE FIRST
045D BF0100                       MOV     DI,1
0460 E85F00                       CALL    INIT_DRV_S
0463 724D                         JC      B3
0465 BF0000                       MOV     DI,0
0468 E85700                       CALL    INIT_DRV_S
046B 7245                         JC      B3
046D BF0200                       MOV     DI,2
0470 E84F00                       CALL    INIT_DRV_S
0473 723D                         JC      B3
0475 BF0400                       MOV     DI,4
0478 E84700                       CALL    INIT_DRV_S
047B 7235                         JC      B3
047D BF0300                       MOV     DI,3
0480 E83F00                       CALL    INIT_DRV_S
0483 722D                         JC      B3
0485 BF0600                       MOV     DI,6
0488 E83700                       CALL    INIT_DRV_S
048B 7225                         JC      B3
```

```
LOC OBJ                 SOURCE

048D BF0500                     MOV     DI,5
0490 E82F00                     CALL    INIT_DRV_S
0493 7210                       JC      B3

0495 BF0700                     MOV     DI,7
0498 E82700                     CALL    INIT_DRV_S
049B 7215                       JC      B3

049D BF0800                     MOV     DI,8                    ; DRIVE STEP OPTION
04A0 268A01                     MOV     AL,ES:[BX + DI]
04A3 A27600                     MOV     CONTROL_BYTE,AL

04A6 2BC9                       SUB     CX,CX
04A8                    B5:
04A8 E8D302                     CALL    PORT_1
04AB EC                         IN      AL,DX
04AC A802                       TEST    AL,R1_IOMODE            ; STATUS INPUT MODE
04AE 7509                       JNZ     B6
04B0 E2F6                       LOOP    B5
04B2                    B3:
04B2 C606740007                 MOV     DISK_STATUS,INIT_FAIL   ; OPERATION FAILED
04B7 F9                         STC
04B8 C3                         RET

04B9                    B6:
04B9 E8B502                     CALL    PORT_0
04BC EC                         IN      AL,DX
04BD 2402                       AND     AL,2                    ; MASK ERROR BIT
04BF 75F1                       JNZ     B3
04C1 C3                         RET
                                ASSUME  ES:NOTHING
                        INIT_DRV_R      ENDP

                        ;----- SEND THE BYTE OUT TO THE CONTROLLER

04C2                    INIT_DRV_S      PROC    NEAR
04C2 E8C501                     CALL    HD_WAIT_REQ
04C5 7207                       JC      D1
04C7 E8A702                     CALL    PORT_0
04CA 268A01                     MOV     AL,ES:[BX + DI]
04CD EE                         OUT     DX,AL
04CE                    D1:
04CE C3                         RET
                        INIT_DRV_S      ENDP

                        ;----------------------------------------
                        ;       READ LONG  (AH = 0AH)           :
                        ;----------------------------------------

04CF                    RD_LONG         PROC    NEAR
04CF E81900                     CALL    CHK_LONG
04D2 726B                       JC      G8
04D4 C6064200E5                 MOV     CMD_BLOCK+0,RD_LONG_CMD
04D9 B047                       MOV     AL,DMA_READ
04DB EB68                       JMP     SHORT   DMA_OPN
                        RD_LONG         ENDP

                        ;----------------------------------------
                        ;       WRITE LONG  (AH = 0BH)          :
                        ;----------------------------------------

04DD                    WR_LONG         PROC    NEAR
04DD E80B00                     CALL    CHK_LONG
04E0 7250                       JC      G8
04E2 C6064200E6                 MOV     CMD_BLOCK+0,WR_LONG_CMD
04E7 B04B                       MOV     AL,DMA_WRITE
04E9 EB5A                       JMP     SHORT   DMA_OPN
                        WR_LONG         ENDP

04EB                    CHK_LONG        PROC    NEAR
04EB A04600                     MOV     AL,CMD_BLOCK+4
04EE 3C80                       CMP     AL,080H
04F0 F5                         CMC
04F1 C3                         RET
                        CHK_LONG        ENDP
```

```
LOC OBJ                SOURCE

                       ;-----------------------------------------
                       ;       SEEK   (AH = 0CH)                :
                       ;-----------------------------------------

04F2                   DISK_SEEK       PROC    NEAR
04F2 C60642000B                MOV     CMD_BLOCK,SEEK_CMD
04F7 EB3D                      JMP     SHORT   NDMA_OPN
                       DISK_SEEK       ENDP

                       ;-----------------------------------------------
                       ;       READ SECTOR BUFFER    (AH = 0EH)        :
                       ;-----------------------------------------------

04F9                   RD_BUFF PROC    NEAR
04F9 C60642000E                MOV     CMD_BLOCK+0,RD_BUFF_CMD
04FE C606460001                MOV     CMD_BLOCK+4,1           ; ONLY ONE BLOCK
0503 B047                      MOV     AL,DMA_READ
0505 EB3E                      JMP     SHORT   DMA_OPN
                       RD_BUFF ENDP

                       ;-----------------------------------------------
                       ;       WRITE SECTOR BUFFER    (AH = 0FH)       :
                       ;-----------------------------------------------

0507                   WR_BUFF PROC    NEAR
0507 C60642000F                MOV     CMD_BLOCK+0,WR_BUFF_CMD
050C C606460001                MOV     CMD_BLOCK+4,1           ; ONLY ONE BLOCK
0511 B04B                      MOV     AL,DMA_WRITE
0513 EB30                      JMP     SHORT   DMA_OPN
                       WR_BUFF ENDP

                       ;-----------------------------------------------
                       ;       TEST DISK READY    (AH = 010H)          :
                       ;-----------------------------------------------

0515                   TST_RDY PROC    NEAR
0515 C606420000                MOV     CMD_BLOCK+0,TST_RDY_CMD
051A EB1A                      JMP     SHORT   NDMA_OPN
                       TST_RDY ENDP

                       ;-----------------------------------------------
                       ;       RECALIBRATE    (AH = 011H)              :
                       ;-----------------------------------------------

051C                   HDISK_RECAL     PROC    NEAR
051C C606420001                MOV     CMD_BLOCK,RECAL_CMD
0521 EB13                      JMP     SHORT   NDMA_OPN
                       HDISK_RECAL     ENDP

                       ;-------------------------------------------------------
                       ;       CONTROLLER RAM DIAGNOSTICS    (AH = 012H)       :
                       ;-------------------------------------------------------

0523                   RAM_DIAG        PROC    NEAR
0523 C6064200E0                MOV     CMD_BLOCK+0,RAM_DIAG_CMD
0528 EB0C                      JMP     SHORT   NDMA_OPN
                       RAM_DIAG        ENDP

                       ;-----------------------------------------------
                       ;       DRIVE DIAGNOSTICS    (AH = 013H)        :
                       ;-----------------------------------------------

052A                   CHK_DRV PROC    NEAR
052A C6064200E3                MOV     CMD_BLOCK+0,CHK_DRV_CMD
052F EB05                      JMP     SHORT   NDMA_OPN
                       CHK_DRV ENDP

                       ;-----------------------------------------------------------
                       ;      CONTROLLER INTERNAL DIAGNOSTICS    (AH = 014H)       :
                       ;-----------------------------------------------------------

0531                   CNTLR_DIAG      PROC    NEAR
0531 C6064200E4                MOV     CMD_BLOCK+0,CNTLR_DIAG_CMD
                       CNTLR_DIAG      ENDP
```

Appendix A

```
LOC OBJ                  SOURCE

                         ;-------------------------------------------------------
                         ;                  SUPPORT ROUTINES                    :
                         ;-------------------------------------------------------

0536                     NDMA_OPN:
0536 B002                        MOV     AL,02H
0538 E82700                      CALL    COMMAND                 ; ISSUE THE COMMAND
053B 7221                        JC      G11
053D EB16                        JMP     SHORT   G3
053F                     G8:
053F C606740009                  MOV     DISK_STATUS,DMA_BOUNDARY
0544 C3                          RET
0545                     DMA_OPN:
0545 E85701                      CALL    DMA_SETUP               ; SET UP FOR DMA OPERATION
0548 72F5                        JC      G8
054A B003                        MOV     AL,03H
054C E81300                      CALL    COMMAND                 ; ISSUE THE COMMAND
054F 720D                        JC      G11
0551 B003                        MOV     AL,03H
0553 E60A                        OUT     DMA+10,AL               ; INITIALIZE THE DISK CHANNEL
0555                     G3:
0555 E421                        IN      AL,021H
0557 24DF                        AND     AL,0DFH
0559 E621                        OUT     021H,AL
055B E8AA01                      CALL    WAIT_INT
055E                     G11:
055E E83B00                      CALL    ERROR_CHK
0561 C3                          RET

                         ;-------------------------------------------------------
                         ; COMMAND                                              :
                         ;       THIS ROUTINE OUTPUTS THE COMMAND BLOCK         :
                         ; INPUT                                                :
                         ;       AL = CONTROLLER DMA/INTERRUPT REGISTER MASK    :
                         ;                                                      :
                         ;-------------------------------------------------------

0562                     COMMAND PROC    NEAR
0562 BE4200                      MOV     SI,OFFSET CMD_BLOCK
0565 E81B02                      CALL    PORT_2
0568 EE                          OUT     DX,AL                   ; CONTROLLER SELECT PULSE
0569 E81C02                      CALL    PORT_3
056C EE                          OUT     DX,AL
056D 2BC9                        SUB     CX,CX                   ; WAIT COUNT
056F E80C02                      CALL    PORT_1
0572                     WAIT_BUSY:
0572 EC                          IN      AL,DX                   ; GET STATUS
0573 240F                        AND     AL,0FH
0575 3C00                        CMP     AL,R1_BUSY OR R1_BUS OR R1_REQ
0577 7409                        JE      C1
0579 E2F7                        LOOP    WAIT_BUSY
057B C606740080                  MOV     DISK_STATUS,TIME_OUT
0580 F9                          STC
0581 C3                          RET                             ; ERROR RETURN
0582                     C1:
0582 FC                          CLD
0583 B90600                      MOV     CX,6                    ; BYTE COUNT
0586                     CM3:
0586 E8E801                      CALL    PORT_0
0589 AC                          LODSB                           ; GET THE NEXT COMMAND BYTE
058A EE                          OUT     DX,AL                   ; OUT IT GOES
058B E2F9                        LOOP    CM3                     ; DO MORE

058D E8EE01                      CALL    PORT_1                  ; STATUS
0590 EC                          IN      AL,DX
0591 A801                        TEST    AL,R1_REQ
0593 7406                        JZ      CM7
0595 C606740020                  MOV     DISK_STATUS,BAD_CNTLR
059A F9                          STC
059B                     CM7:
059B C3                          RET
                         COMMAND ENDP

                         ;----------------------------------------------
                         ;               SENSE STATUS BYTES            :
                         ;                                             :
                         ; BYTE 0                                      :
```

```
LOC OBJ                 LINE    SOURCE

                                ;         BIT    7   ADDRESS VALID, WHEN SET       :
                                ;         BIT    6   SPARE, SET TO ZERO            :
                                ;         BITS 5-4   ERROR TYPE                    :
                                ;         BITS 3-0   ERROR CODE                    :
                                ;                                                  :
                                ; BYTE 1                                           :
                                ;         BITS 7-6   ZERO                          :
                                ;         BIT    5   DRIVE (0-1)                   :
                                ;         BITS 4-0   HEAD NUMBER                   :
                                ;                                                  :
                                ; BYTE 2                                           :
                                ;         BITS 7-5   CYLINDER HIGH                 :
                                ;         BITS 4-0   SECTOR NUMBER                 :
                                ;                                                  :
                                ; BYTE 3                                           :
                                ;         BITS 7-0   CYLINDER LOW                  :
                                ;                                                  :
                                ;--------------------------------------------------

059C                            ERROR_CHK       PROC    NEAR
                                        ASSUME  ES:DATA
059C A07400                             MOV     AL,DISK_STATUS          ; CHECK IF THERE WAS AN ERROR
059F 0AC0                               OR      AL,AL
05A1 7501                               JNZ     G21
05A3 C3                                 RET

                                ;----- PERFORM SENSE STATUS

05A4                            G21:
05A4 B84000                             MOV     AX,DATA
05A7 8EC0                               MOV     ES,AX                   ; ESTABLISH SEGMENT
05A9 2BC0                               SUB     AX,AX
05AB 8BF8                               MOV     DI,AX
05AD C606420003                         MOV     CMD_BLOCK+0,SENSE_CMD
05B2 2AC0                               SUB     AL,AL
05B4 E8ABFF                             CALL    COMMAND                 ; ISSUE SENSE STATUS COMMAND
05B7 7223                               JC      SENSE_ABORT             ; CANNOT RECOVER
05B9 B90400                             MOV     CX,4
05BC                            G22:
05BC E8CB00                             CALL    HD_WAIT_REQ
05BF 7220                               JC      G24
05C1 E8AD01                             CALL    PORT_0
05C4 EC                                 IN      AL,DX
05C5 26884542                           MOV     ES:HD_ERROR[DI],AL      ; STORE AWAY SENSE BYTES
05C9 47                                 INC     DI
05CA E8B101                             CALL    PORT_1
05CD E2ED                               LOOP    G22
05CF E8B800                             CALL    HD_WAIT_REQ
05D2 720D                               JC      G24
05D4 E89A01                             CALL    PORT_0
05D7 EC                                 IN      AL,DX
05D8 A802                               TEST    AL,2
05DA 740F                               JZ      STAT_ERR
05DC                            SENSE_ABORT:
05DC C6067400FF                         MOV     DISK_STATUS,SENSE_FAIL
05E1                            G24:
05E1 F9                                 STC
05E2 C3                                 RET
                                ERROR_CHK       ENDP

05E3 1A06                       T_0     DW      TYPE_0
05E5 2706                       T_1     DW      TYPE_1
05E7 6A06                       T_2     DW      TYPE_2
05E9 7706                       T_3     DW      TYPE_3

05EB                            STAT_ERR:
05EB 268A1E4200                         MOV     BL,ES:HD_ERROR          ; GET ERROR BYTE
05F0 8AC3                               MOV     AL,BL
05F2 240F                               AND     AL,0FH
05F4 80E330                             AND     BL,00110000B            ; ISOLATE TYPE
05F7 2AFF                               SUB     BH,BH
05F9 B103                               MOV     CL,3
05FB D3EB                               SHR     BX,CL                   ; ADJUST
05FD 2EFFA7E305                         JMP     WORD PTR CS:[BX + OFFSET T_0]
                                        ASSUME  ES:NOTHING

0602                            TYPE0_TABLE     LABEL   BYTE
```

Appendix A

```
LOC OBJ                 LINE     SOURCE

0602 00204020800020              DB      0,BAD_CNTLR,BAD_SEEK,BAD_CNTLR,TIME_OUT,0,BAD_CNTLR
0609 0040                        DB      0,BAD_SEEK
  0009                  TYPE0_LEN        EQU     $-TYPE0_TABLE
060B                    TYPE1_TABLE      LABEL   BYTE
060B 1010020004                  DB      BAD_ECC,BAD_ECC,BAD_ADDR_MARK,0,RECORD_NOT_FND
0610 400000110B                  DB      BAD_SEEK,D,D,DATA_CORRECTED,BAD_TRACK
  000A                  TYPE1_LEN        EQU     $-TTPE1_TABLE
0615                    TYPE2_TABLE      LABEL   BTTE
0615 0102                        DB      BAD_CMD,BAD_ADDR_MARK
  0002                  TYPE2_LEN        EQU     $-TYPE2_TABLE
0617                    TYPE3_TABLE      LABEL   BTTE
0617 202010                      DB      BAD_CNTLR,BAD_CNTLR,BAD_ECC
  0003                  TYPE3_LEN        EQU     $-TYPE3_TABLE

                        ;----- TYPE 0 ERROR

061A                    TYPE_0:
061A BB0206                      MOV     BX,DFFSET TTPE0_TABLE
061D 3C09                        CMP     AL,TYPED_LEN            ; CHECK IF ERROR IS DEFINED
061F 7363                        JAE     UNDEF_ERR_L
0621 2E07                        XLAT    CS:TYPE0_TABLE          ; TABLE LOOKUP
0623 A27400                      MOV     DISK_STATUS,AL          ; SET ERROR CODE
0626 C3                          RET

                        ;------ TYPE 1 ERROR

0627                    TYPE_1:
0627 BB0B06                      MOV     BX,DFFSET TYPE1_TABLE
062A 8BCB                        MOV     CX,AX
062C 3C0A                        CMP     AL,TYPE1_LEN            ; CHECK IF ERROR IS DEFINED
062E 7354                        JAE     UNDEF_ERR_L
0630 2E07                        XLAT    CS:TYPE1_TABLE          ; TABLE LOOKUP
0632 A27400                      MOV     DISK_STATUS,AL          ; SET ERROR CODE
0635 80E108                      AND     CL,D8H                  ; CORRECTED ECC
0638 80F908                      CMP     CL,D8H
063B 752A                        JNZ     630

                        ;----- OBTAIN ECC ERROR BURST LENGTH

063D C606420000                  MOV     CMD_BLOCK+0,RD_ECC_CMD
0642 2AC0                        SUB     AL,AL
0644 E81BFF                      CALL    COMMAND
0647 721E                        JC      630
0649 E83E00                      CALL    HD_WAIT_REQ
064C 7219                        JC      630
064E E82001                      CALL    PORT_0
0651 EC                          IN      AL,DX
0652 8AC8                        MOV     CL,AL
0654 E83300                      CALL    HD_WAIT_REQ
0657 720E                        JC      630
0659 E81501                      CALL    PORT_0
065C EC                          IN      AL,DX
065D A801                        TEST    AL,D1H
065F 7406                        JZ      630
0661 C606740020                  MOV     DISK_STATUS,BAD_CNTLR
0666 F9                          STC
0667                    630:
0667 8AC1                        MOV     AL,CL
0669 C3                          RET

                        ;----- TYPE 2 ERROR

066A                    TYPE_2:
066A BB1506                      MOV     BX,DFFSET TTPE2_TABLE
066D 3C02                        CMP     AL,TYPE2_LEN            ; CHECK IF ERROR IS DEFINED
066F 7313                        JAE     UNDEF_ERR_L
0671 2E07                        XLAT    CS:TTPE1_TABLE          ; TABLE LOOKUP
0673 A27400                      MOV     DISK_STATUS,AL          ; SET ERROR CODE
0676 C3                          RET

                        ;----- TYPE 3 ERROR

0677                    TYPE_3:
0677 BB1706                      MOV     BX,DFFSET TTPE3_TABLE
067A 3C03                        CMP     AL,TYPE3_LEN
067C 7306                        JAE     UNDEF_ERR_L
067E 2E07                        XLAT    CS:TTPE3_TABLE
```

```
LOC OBJ                LINE    SOURCE

0680 A27400                             MOV     DISK_STATUS,AL
0683 C3                                 RET

0684                            UNDEF_ERR_L:
0684 C6067400BB                         MOV     DISK_STATUS,UNDEF_ERR
0689 C3                                 RET

068A                            HD_WAIT_REQ     PROC    NEAR
068A 51                                 PUSH    CX
068B 2BC9                               SUB     CX,CX
068D E8EE00                             CALL    PORT_1
0690                            L1:
0690 EC                                 IN      AL,DX
0691 A801                               TEST    AL,R1_REQ
0693 7508                               JNZ     L2
0695 E2F9                               LOOP    L1
0697 C606740080                         MOV     DISK_STATUS,TIME_OUT
069C F9                                 STC
069D                            L2:
069D 59                                 POP     CX
069E C3                                 RET
                                HD_WAIT_REQ     ENDP

                                ;-------------------------------------------------------
                                ; DMA_SETUP                                            :
                                ;       THIS ROUTINE SETS UP FOR DMA OPERATIONS.       :
                                ; INPUT                                                :
                                ;       (AL) = MODE BYTE FOR THE DMA                   :
                                ;       (ES:BX) = ADDRESS TO READ/WRITE THE DATA       :
                                ; OUTPUT                                               :
                                ;       (AX) DESTROYED                                 :
                                ;-------------------------------------------------------
069F                            DMA_SETUP       PROC    NEAR
069F 50                                 PUSH    AX
06A0 A04600                             MOV     AL,CMD_BLOCK+4
06A3 3C81                               CMP     AL,81H                  ; BLOCK COUNT OUT OF RANGE
06A5 58                                 POP     AX
06A6 7202                               JB      J1
06A8 F9                                 STC
06A9 C3                                 RET
06AA                            J1:
06AA 51                                 PUSH    CX                      ; SAVE THE REGISTER
06AB FA                                 CLI                             ; NO MORE INTERRUPTS
06AC E60C                               OUT     DMA+12,AL               ; SET THE FIRST/LAST F/F
06AE 50                                 PUSH    AX
06AF 58                                 POP     AX
06B0 E60B                               OUT     DMA+11,AL               ; OUTPUT THE MODE BYTE
06B2 8CC0                               MOV     AX,ES                   ; GET THE ES VALUE
06B4 B104                               MOV     CL,4                    ; SHIFT COUNT
06B6 D3C0                               ROL     AX,CL                   ; ROTATE LEFT
06B8 8AE8                               MOV     CH,AL                   ; GET HIGHEST NYBBLE OF ES TO CH
06BA 24F0                               AND     AL,0F0H                 ; ZERO THE LOW NYBBLE FROM SEGMENT
06BC 03C3                               ADD     AX,BX                   ; TEST FOR CARRY FROM ADDITION
06BE 7302                               JNC     J33
06C0 FEC5                               INC     CH                      ; CARRY MEANS HIGH 4 BITS MUST BE INC
06C2                            J33:
06C2 50                                 PUSH    AX                      ; SAVE START ADDRESS
06C3 E606                               OUT     DMA+6,AL                ; OUTPUT LOW ADDRESS
06C5 8AC4                               MOV     AL,AH
06C7 E606                               OUT     DMA+6,AL                ; OUTPUT HIGH ADDRESS
06C9 8AC5                               MOV     AL,CH                   ; GET HIGH 4 BITS
06CB 240F                               AND     AL,0FH
06CD E682                               OUT     DMA_HIGH,AL             ; OUTPUT THE HIGH 4 BITS TO PAGE REG

                                ;------ DETERMINE COUNT

06CF A04600                             MOV     AL,CMD_BLOCK+4          ; RECOVER BLOCK COUNT
06D2 D0E0                               SHL     AL,1                    ; MULTIPLY BY 512 BYTES PER SECTOR
06D4 FEC8                               DEC     AL                      ;   AND DECREMENT VALUE BY ONE
06D6 8AE0                               MOV     AH,AL
06D8 B0FF                               MOV     AL,0FFH

                                ;----- HANDLE READ AND WRITE LONG (516D BYTE BLOCKS)

06DA 50                                 PUSH    AX                      ; SAVE REGISTER
06DB A04200                             MOV     AL,CMD_BLOCK+0          ; GET COMMAND
06DE 3CE5                               CMP     AL,RD_LONG_CMD
```

Appendix A

```
LOC OBJ              SOURCE

06E0 7407                     JE      AD04
06E2 3CE6                     CMP     AL,WR_LONG_CMD
06E4 7403                     JE      AD04
06E6 58                       POP     AX                      ; RESTORE REGISTER
06E7 EB11                     JMP     SHORT   J20
06E9                  AD04:
06E9 58                       POP     AX                      ; RESTORE REGISTER
06EA B80402                   MOV     AX,516D                 ; ONE BLOCK (512) PLUS 4 BYTES ECC
06ED 53                       PUSH    BX
06EE 2AFF                     SUB     BH,BH
06F0 8A1E4600                 MOV     BL,CMD_BLOCK+4
06F4 52                       PUSH    DX
06F5 F7E3                     MUL     BX                      ; BLOCK COUNT TIMES 516
06F7 5A                       POP     DX
06F8 5B                       POP     BX
06F9 48                       DEC     AX                      ; ADJUST
06FA                  J20:

06FA 50                       PUSH    AX                      ; SAVE COUNT VALUE
06FB E607                     OUT     DMA+7,AL                ; LOW BYTE OF COUNT
06FD 8AC4                     MOV     AL,AH
06FF E607                     OUT     DMA+7,AL                ; HIGH BYTE OF COUNT
0701 FB                       STI                             ; INTERRUPTS BACK ON
0702 59                       POP     CX                      ; RECOVER COUNT VALUE
0703 58                       POP     AX                      ; RECOVER ADDRESS VALUE
0704 03C1                     ADD     AX,CX                   ; ADD, TEST FOR 64K OVERFLOW
0706 59                       POP     CX                      ; RECOVER REGISTER
0707 C3                       RET                     ; RETURN TO CALLER, CFL SET BY ABOVE IF ERROR
                      DMA_SETUP       ENDP

                      ;----------------------------------------------
                      ; WAIT_INT                                     :
                      ;       THIS ROUTINE WAITS FOR THE FIXED DISK  :
                      ;       CONTROLLER TO SIGNAL THAT AN INTERRUPT :
                      ;       HAS OCCURRED.                          :
                      ;----------------------------------------------
0708                  WAIT_INT        PROC    NEAR
0708 FB                       STI                             ; TURN ON INTERRUPTS
0709 53                       PUSH    BX                      ; PRESERVE REGISTERS
070A 51                       PUSH    CX
070B 06                       PUSH    ES
070C 56                       PUSH    SI
070D 1E                       PUSH    DS
                              ASSUME  DS:DUMMY
070E 2BC0                     SUB     AX,AX
0710 8ED8                     MOV     DS,AX                   ; ESTABLISH SEGMENT
0712 C4360401                 LES     SI,HF_TBL_VEC
                              ASSUME  DS:DATA
0716 1F                       POP     DS

                      ;----- SET TIMEOUT VALUES

0717 2AFF                     SUB     BH,BH
0719 268A5C09                 MOV     BL,BYTE PTR ES:[SI][9]          ; STANDARD TIME OUT
071D 8A264200                 MOV     ~
0721 80FC04                   CMP     AH,FMTDRV_CMD
0724 7506                     JNZ     W5
0726 268A5C0A                 MOV     BL,BYTE PTR ES:[SI][0AH]        ; FORMAT DRIVE
072A EB09                     JMP     SHORT   W4
072C 80FCE3           W5:     CMP     AH,CHK_DRV_CMD
072F 7504                     JNZ     W4
0731 268A5C0B                 MOV     BL,BYTE PTR ES:[SI][0BH]        ; CHECK DRIVE
0735                  W4:
0735 2BC9                     SUB     CX,CX

                      ;----- WAIT FOR INTERRUPT

0737                  W1:
0737 E84400                   CALL    PORT_1
073A EC                       IN      AL,DX
073B 2420                     AND     AL,020H
073D 3C20                     CMP     AL,020H                 ; DID INTERRUPT OCCUR
073F 740A                     JZ      W2
0741 E2F4                     LOOP    W1                      ; INNER LOOP
0743 4B                       DEC     BX
0744 75F1                     JNZ     W1                      ; OUTER LOOP
0746 C606740080               MOV     DISK_STATUS,TIME_OUT
074B                  W2:
```

```
LOC OBJ              SOURCE

074B E82300                 CALL   PORT_0
074E EC                     IN     AL,DX
074F 2402                   AND    AL,2                     ; ERROR BIT
0751 08067400               OR     DISK_STATUS,AL           ; SAVE
0755 E83000                 CALL   PORT_3                   ; INTERRUPT MASK REGISTER
0758 32C0                   XOR    AL,AL                    ; ZERO
075A EE                     OUT    DX,AL                    ; RESET MASK
075B 5E                     POP    SI                       ; RESTORE REGISTERS
075C 07                     POP    ES
075D 59                     POP    CX
075E 5B                     POP    BX
075F C3                     RET
                     WAIT_INT       ENDP

0760                 HD_INT  PROC   NEAR
0760 50                     PUSH   AX
0761 B020                   MOV    AL,EOI                   ; END OF INTERRUPT
0763 E620                   OUT    INT_CTL_PORT,AL
0765 B007                   MOV    AL,07H                   ; SET DMA MODE TO DISABLE
0767 E60A                   OUT    DMA+10,AL
0769 E421                   IN     AL,021H
076B 0C20                   OR     AL,020H
076D E621                   OUT    021H,AL
076F 58                     POP    AX
0770 CF                     IRET
                     HD_INT  ENDP

                     ;----------------------------------------
                     ; PORTS                                  :
                     ;       GENERATE PROPER PORT VALUE       :
                     ;       BASED ON THE PORT OFFSET         :
                     ;----------------------------------------

0771                 PORT_0  PROC   NEAR
0771 BA2003                 MOV    DX,HF_PORT               ; BASE VALUE
0774 50                     PUSH   AX
0775 2AE4                   SUB    AH,AH
0777 A07700                 MOV    AL,PORT_OFF              ; ADD IN THE OFFSET
077A 03D0                   ADD    DX,AX
077C 58                     POP    AX
077D C3                     RET
                     PORT_0  ENDP

077E                 PORT_1  PROC   NEAR
077E E8F0FF                 CALL   PORT_0
0781 42                     INC    DX                       ; INCREMENT TO PORT ONE
0782 C3                     RET
                     PORT_1  ENDP

0783                 PORT_2  PROC   NEAR
0783 E8F8FF                 CALL   PORT_1
0786 42                     INC    DX                       ; INCREMENT TO PORT TWO
0787 C3                     RET
                     PORT_2  ENDP

0788                 PORT_3  PROC   NEAR
0788 E8F8FF                 CALL   PORT_2
078B 42                     INC    DX                       ; INCREMENT TO PORT THREE
078C C3                     RET
                     PORT_3  ENDP

                     ;------------------------------------------------
                     ; SW2_OFFS                                      :
                     ;       DETERMINE PARAMETER TABLE OFFSET        :
                     ;       USING CONTROLLER PORT TWO AND           :
                     ;       DRIVE NUMBER SPECIFIER (0-1)            :
                     ;------------------------------------------------

078D                 SW2_OFFS       PROC    NEAR
078D E8F3FF                 CALL   PORT_2
0790 EC                     IN     AL,DX                    ; READ PORT 2
0791 50                     PUSH   AX
0792 E8E9FF                 CALL   PORT_1
0795 EC                     IN     AL,DX
0796 2402                   AND    AL,2                     ; CHECK FOR ERROR
0798 58                     POP    AX
0799 7516                   JNZ    SW2_OFFS_ERR
079B 8A264300               MOV    AH,CMD_BLOCK+1
```

Appendix A

```
LOC OBJ                 SOURCE

079F 80E420                     AND     AH,00100000B            ; DRIVE 0 OR 1
07A2 7504                       JNZ     SW2_AND
07A4 D0E8                       SHR     AL,1                    ; ADJUST
07A6 D0E8                       SHR     AL,1
07A8                    SW2_AND:
07A8 2403                       AND     AL,011B                 ; ISOLATE
07AA B104                       MOV     CL,4
07AC D2E0                       SHL     AL,CL                   ; ADJUST
07AE 2AE4                       SUB     AH,AH
07B0 C3                         RET
07B1                    SW2_OFFS_ERR:
07B1 F9                         STC
07B2 C3                         RET
                        SW2_OFFS        ENDP

07B3 30382F31362F38             DB      '08/16/82'              ; RELEASE MARKER
     32

07BB                    END_ADDRESS     LABEL   BYTE
----                    CODE    ENDS
                                END
```

APPENDIX B: 8088 ASSEMBLY INSTRUCTION SET REFERENCE

8088 Register Model

AX:	AH	AL	Accumulator	General Register File
BX:	BH	BL	Base	
CX:	CH	CL	Count	
DX:	DH	DL	Data	
	SP		Stack Pointer	
	BP		Base Pointer	
	SI		Source Index	
	DI		Destination Index	
	IP		Instruction Pointer	
	FLAGSH	FLAGSL	Status Flags	
	CS		Code Segment	Segment Register File
	DS		Data Segment	
	SS		Stack Segment	
	ES		Extra Segment	

Instructions which reference the flag register file as a 16-bit object use the symbol FLAGS to represent the file:

15								7							0
X	X	X	X	OF	DF	IF	TF	SF	ZF	X	AF	X	PF	X	CF

x = Don't Care

AF:	Auxiliary Carry - BCD	8080 Flags
CF:	Carry Flag	
PF:	Parity Flag	
SF:	Sign Flag	
ZF:	Zero Flag	

DF:	Direction Flag (Strings)	8088 Flags
IF:	Interrupt Enable Flag	
OF:	Overflow Flag (CF $\oplus$ SF)	
TF:	Trap - Single Step Flag	

Operand Summary

"reg field Bit Assignments:

16-Bit (w=1)	8-Bit (w=0)	Segment
000 AX	000 AL	00 ES
001 CX	001 CL	01 CS
010 DX	010 DL	10 SS
011 BX	011 BL	11 DS
100 SP	100 AH	
101 BP	101 CH	
110 SI	110 DH	
111 DI	111 BH	

Second Instruction Byte Summary

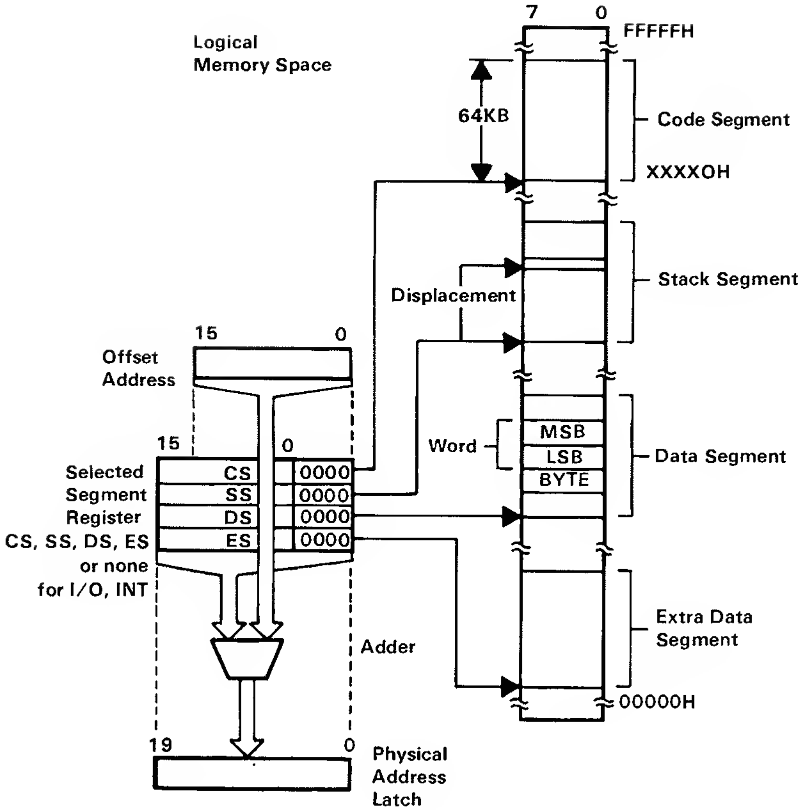
mod	xxx	r/m

mod	Displacement
00	DISP=0*, disp-low and disp-high are absent
01	DISP=disp-low sign-extended to 16-bits, disp-high is absent
10	DISP=disp-high: disp-low
11	r/m is treated as a "reg" field

r/m	Operand Address
000	(BX) + (SI) + DISP
001	(BX) + (DI) + DISP
010	(BP) + (SI) + DISP
011	(BP) + (DI) + DISP
100	(SI) + DISP
101	(DI) + DISP
110	(BP) + DISP*
111	(BX) + DISP

DISP follows 2nd byte of instruction (before data if required).
*except if mod = 00 and r/m = 110 then EA = disp-high: disp-low.

Memory Segmentation Model



Segment Override Prefix

0 0 1 reg 1 1 0

Use of Segment Override

Operand Register	Default	With Override Prefix
IP (Code Address)	CS	Never
SP (Stack Address)	SS	Never
BP (Stack Address or Stack Marker)	SS	BP + DS or ES, or CS
SI or DI (not including strings)	DS	ES, SS, or CS
SI (Implicit Source Address for Strings)	DS	ES, SS, or CS
DI (Implicit Destination Address for Strings)	ES	Never

MOV = Move

Register/memory to/from register

1	0	0	0	1	0	d	w	mod reg r/m

Immediate to register/memory

1	1	0	0	0	1	1	w	mod 0 0 0 r/m	data	data if w=1

Immediate to register

1	0	1	1	w	reg	data	data if w=1

Memory to accumulator

1	0	1	0	0	0	0	w	addr-low	addr-high

Accumulator to memory

1	0	1	0	0	0	1	w	addr-low	addr-high

Register/memory to segment register

1	0	0	0	1	1	1	0	mod 0 reg r/m

Segment register to register/memory

1	0	0	0	1	1	0	0	mod 0 reg r/m

PUSH = Push

Register/memory

1	1	1	1	1	1	1	1	mod 1 1 0 r/m

Register

0	1	0	1	0	reg

Segment register

0	0	0	reg	1	1	0

POP = Pop

Register/memory

1	0	0	0	1	1	1	1	mod 0 0 0 r/m

Register

0	1	0	1	1	reg

Segment register

0	0	0	reg	1	1	1

Appendix B

XCHG = Exchange
Register/memory with register

1	0	0	0	0	1	1	w	mod reg r/m

Register with accumulator

1	0	0	1	0	reg

IN = Input to AL/AX from
Fixed port

1	1	1	0	0	1	0	w	port

Variable port (DX)

1	1	1	0	1	1	0	w

OUT = Output from AL/AX to
Fixed port

1	1	1	0	0	1	1	w	port

Variable port (DX)

1	1	1	0	1	1	0	w

XLAT = Translate byte to AL

1	1	0	1	0	1	1	1

LEA = Load EA to register

1	0	0	0	1	1	0	1	mod reg r/m

LDS = Load pointer to DS

1	1	0	0	0	1	0	1	mod reg r/m

LES = Load pointer to ES

1	1	0	0	0	1	0	0	mod reg r/m

LAHF = Load AH with flags

1	0	0	1	1	1	1	1

SAHF = Store AH into flags

1	0	0	1	1	1	1	0

PUSHF = Push flags

1	0	0	1	1	1	0	0

POPF = Pop flags

1	0	0	1	1	1	0	1

Arithmetic

ADD = Add
Register/memory with register to either

0	0	0	0	0	0	d	w	mod reg r/m

Immediate to register/memory

1	0	0	0	0	0	s	w	mod 0 0 0 r/m	data	data if s:w=01

Immediate to accumulator

0	0	0	0	0	1	0	w	data	data if w=1

ADC = Add with carry
Register/memory with register to either

0	0	0	1	0	0	d	w	mod reg r/m

Immediate to register/memory

1	0	0	0	0	0	s	w	mod 0 1 0 r/m	data	data if s:w=01

Immediate to accumulator

0	0	0	1	0	1	0	w	data	data if w=1

INC = Increment
Register/memory

1	1	1	1	1	1	1	w	mod 0 0 0 r/m

Register

0	1	0	0	0	reg

AAA = ASCII adjust for add

0	0	1	1	0	1	1	1

DAA = Decimal adjust for add

0	0	1	0	0	1	1	1

SUB = Subtract
Register/memory and register to either

0	0	1	0	1	0	d	w	mod reg r/m

Immediate from register/memory

1	0	0	0	0	0	s	w	mod 1 0 1 r/m	data	data if s:w=01

Immediate from accumulator

0	0	1	0	1	1	0	w	data	data if w=1

Appendix B

SBB = Subtract with borrow
Register/memory and register to either

0	0	0	1	1	0	d	w	mod reg r/m

Immediate from register/memory

1	0	0	0	0	0	s	w	mod 0 1 1 r/m	data	data if s:w=01

Immediate from accumulator

0	0	0	1	1	1	0	w	data	data if w=1

DEC = Decrement
Register/memory

1	1	1	1	1	1	1	w	mod 0 0 1 r/m

Register

0	1	0	0	1	reg

NEG = Change sign

1	1	1	1	0	1	1	w	mod 0 1 1 r/m

CMP = Compare
Register/memory and register

0	0	1	1	1	0	d	w	mod reg r/m

Immediate with register/memory

1	0	0	0	0	0	s	w	mod 1 1 1 r/m	data	data if s:w=01

Immediate with accumulator

0	0	1	1	1	1	0	w	data	data if w=1

AAS = ASCII adjust for subtract

0	0	1	1	1	1	1	1

DAS = Decimal adjust for subtract

0	0	1	0	1	1	1	1

MUL = Multiply (unsigned)

1	1	1	1	0	1	1	w	mod 1 0 0 r/m

IMUL = Integer multiply (signed)

1	1	1	1	0	1	1	w	mod 1 0 1 r/m

AAM = ASCII adjust for multiply

1	1	0	1	0	1	0	0	0 0 0 0 1 0 1 0

DIV = Divide (unsigned)

1	1	1	1	0	1	1	w	mod 1 1 0 r/m

IDIV = Integer divide (signed)

1	1	1	1	0	1	1	w	mod 1 1 1 r/m

AAD = ASCII adjust for divide

1	1	0	1	0	1	0	1	0 0 0 0 1 0 1 0

CBW = Convert byte to word

1	0	0	1	1	0	0	0

CWD = Convert word to double word

1	0	0	1	1	0	0	1

Logic

NDT = Invert

1	1	1	1	0	1	1	w	mod 0 1 0 r/m

SHL/SAL = Shift logical/arithmetic left

1	1	0	1	0	0	v	w	mod 1 0 0 r/m

SHR = Shift logical right

1	1	0	1	0	0	v	w	mod 1 0 1 r/m

SAR = Shift arithmetic right

1	1	0	1	0	0	v	w	mod 1 1 1 r/m

ROL = Rotate left

1	1	0	1	0	0	v	w	mod 0 0 0 r/m

RDR = Rotate right

1	1	0	1	0	0	v	w	mod 0 0 1 r/m

RCL = Rotate through carry left

1	1	0	1	0	0	v	w	mod 0 1 0 r/m

RCR = Rotate through carry right

1	1	0	1	0	0	v	w	mod 0 1 1 r/m

AND = And
Register/memory and register to either

0	0	1	0	0	0	d	w	mod reg r/m

Immediate to register/memory

1	0	0	0	0	0	0	w	mod 1 0 0 r/m	data	data if w=1

Immediate to accumulator

0	0	1	0	0	1	0	w	data	data if w=1

Appendix B

TEST = And function to flags, no result
Register/memory and register

1 0 0 0 0 1 0 w	mod reg r/m

Immediate data and register/memory

1 1 1 1 0 1 1 w	mod 0 0 0 r/m	data	data if w=1

Immediate data and accumulator

1 0 1 0 1 0 0 w	data	data if w=1

OR = OR
Register/memory and register to either

0 0 0 0 1 0 d w	mod reg r/m

Immediate to register/memory

1 0 0 0 0 0 0 w	mod 0 0 1 r/m	data	data if w=1

Immediate to accumulator

0 0 0 0 1 1 0 w	data	data if w=1

XOR = Exclusive or
Register/memory and register to either

0 0 1 1 0 0 d w	mod reg r/m

Immediate to register/memory

1 0 0 0 0 0 0 w	mod 1 1 0 r/m	data	data if w=1

Immediate to accumulator

0 0 1 1 0 1 0 w	data	data if w=1

String Manipulation

REP = Repeat

1 1 1 1 0 0 1 z

MOVS = Move String

1 0 1 0 0 1 0 w

CMPS = Compare String

1 0 1 0 0 1 1 w

SCAS = Scan String

1 0 1 0 1 1 1 w

LODS = Load String

1	0	1	0	1	1	0	w

STOS = Store String

1	0	1	0	1	0	1	w

Control Transfer

CALL = Call
Direct within segment

1 1 1 0 1 0 0 0	disp-low	disp-high

Indirect within segment

1 1 1 1 1 1 1 1	mod 0 1 0 r/m

Direct intersegment

1 0 0 1 1 0 1 0	offset-low	offset-high
	seg-low	seg-high

Indirect intersegment

1 1 1 1 1 1 1 1	mod 0 1 1 r/m

JMP = Unconditional Jump
Direct within segment

1 1 1 0 1 0 0 1	disp-low	disp-high

Direct within segment-short

1 1 1 0 1 0 1 1	disp

Indirect within segment

1 1 1 1 1 1 1 1	mod 1 0 0 r/m

Direct intersegment

1 1 1 0 1 0 1 0	offset-low	offset-high
	seg-low	seg-high

Indirect intersegment

1 1 1 1 1 1 1 1	mod 1 0 1 r/m

Appendix B

RET = Return from CALL
Within segment

1	1	0	0	0	0	1	1

Within segment adding immediate to SP

1 1 0 0 0 0 1 0	data-low	data-high

Intersegment

1	1	0	0	1	0	1	1

Intersegment, adding immediate to SP

1 1 0 0 0 0 1 0	data-low	data-high

JE/JZ = Jump on equal/zero

0 1 1 1 0 1 0 0	disp

JL/JNGE = Jump on less/not greater or equal

0 1 1 1 1 1 0 0	disp

JLE/JNG = Jump on less or equal/not greater

0 1 1 1 1 1 1 0	disp

JB/JNAE = Jump on below/not above or equal

0 1 1 1 0 0 1 0	disp

JBE/JNA = Jump on below or equal/not above

0 1 1 1 0 1 1 0	disp

JP/JPE = Jump on parity/parity even

0 1 1 1 1 0 1 0	disp

JO = Jump on overflow

0 1 1 1 0 0 0 0	disp

JS = Jump on sign

0 1 1 1 1 0 0 0	disp

JNE/JNZ = Jump on not equal/not zero

0 1 1 1 0 1 0 1	disp

JNL/JGE = Jump on not less/greater or equal

0 1 1 1 1 1 0 1	disp

JNLE/JG = Jump on not less or equal/greater

0	1	1	1	1	1	1	1	disp

JNB/JAE = Jump on not below/above or equal

0	1	1	1	0	0	1	1	disp

JNBE/JA = Jump on not below or equal/above

0	1	1	1	0	1	1	1	disp

JNP/JPO = Jump on not parity/parity odd

0	1	1	1	1	0	1	1	disp

JNO = Jump on not overflow

0	1	1	1	0	0	0	1	disp

JNS = Jump on not sign

0	1	1	1	1	0	0	1	disp

LOOP = Loop CX times

1	1	1	0	0	0	1	0	disp

LOOPZ/LOOPE = Loop while zero/equal

1	1	1	0	0	0	0	1	disp

LOOPNZ/LOOPNE = Loop while not zero/not equal

1	1	1	0	0	0	0	0	disp

JCXZ = Jump on CX zero

1	1	1	0	0	0	1	1	disp

8088 Conditional Transfer Operations

Instruction	Condition	Interpretation
JE or JZ	ZF = 1	"equal" or "zero"
JL or JNGE	(SF xor OF) = 1	"less" or "not greater or equal"
JLE or JNG	((SF xor OF) or ZF) = 1	"less or equal" or "not greater"
JB or JNAE or JC	CF = 1	"below" or "not above or equal"
JBE or JNA	(CF or ZF) = 1	"below or equal" or "not above"
JP or JPE	PF = 1	"parity" or "parity even"
JO	OF = 1	"overflow"
JS	SF = 1	"sign"
JNE or JNZ	ZF = 0	"not equal" or "not zero"
JNL or JGE	(SF xor OF) = 0	"not less" or "greater or equal"
JNLE or JG	((SF xor OF) or ZF) = 0	"not less or equal" or "greater"
JNB or JAE or JNC	CF = 0	"not below" or "above or equal"
JNBE or JA	(CF or ZF) = 0	"not below or equal" or "above"
JNP or JPO	PF = 0	"not parity" or "parity odd"
JNO	OF = 0	"not overflow"
JNS	SF = 0	"not sign"

*"Above" and "below" refer to the relation between two unsigned values, while "greater" and "less" refer to the relation between two signed values.

INT = Interrupt
Type specified

1	1	0	0	1	1	0	1	type

Type 3

1	1	0	0	1	1	0	0

INTO = Interrupt on overflow

1	1	0	0	1	1	1	0

IRET = Interrupt return

1	1	0	0	1	1	1	1

Processor Control

CLC = Clear carry

1	1	1	1	1	0	0	0

STC = Set carry

1	1	1	1	1	0	0	1

CMC = Complement carry

1	1	1	1	0	1	0	1

NOP = No operation

1	0	0	1	0	0	0	0

CLD = Clear direction

1	1	1	1	1	1	0	0

STD = Set direction

1	1	1	1	1	1	0	1

CLI = Clear interrupt

1	1	1	1	1	0	1	0

STI = Set interrupt

1	1	1	1	1	0	1	1

HLT = Halt

1	1	1	1	0	1	0	0

WAIT = Wait

1	0	0	1	1	0	1	1

LOCK = Bus lock prefix

1	1	1	1	0	0	0	0

ESC = Escape (to external device)

1	1	0	1	1	x	x	x	mod	x	x	x	r/m

Footnotes:
if d = 1 then "to"; if d = 0 then "from"
if w = 1 then word instruction; if w = 0 then byte instruction
if s:w = 01 then 16 bits of immediate data from the operand
if s:w = 11 then an immediate data byte is sign extended to form the 16-bit operand
if v = 0 then "count" = 1; if v = 1 then "count" in (CL)
x = don't care
z is used for some string primitives to compare with ZF FLAG
AL = 8-bit accumulator
AX = 16-bit accumulator
CX = Count register
DS = Data segment
DX = Variable port register
ES = Extra segment
Above/below refers to unsigned value
Greater = more positive;
Less = less positive (more negative) signed values

8088 Instruction Set Matrix

HI \ LO	0	1	2	3	4	5	6	7
0	ADD b,f,r/m	ADD w,f,r/m	ADD b,t,r/m	ADD w,t,r/m	ADD b,ia	ADD w,ia	PUSH ES	POP ES
1	ADC b,f,r/m	ADC w,f,r/m	ADC b,t,r/m	ADC w,t,r/m	ADC b,i	ADC w,i	PUSH SS	POP SS
2	AND b,f,r/m	AND w,f,r/m	AND b,t,r/m	AND w,t,r/m	AND b,i	AND w,i	SEG =ES	DAA
3	XOR b,f,r/m	XOR w,f,r/m	XOR b,t,r/m	XOR w,t,r/m	XOR b,i	XOR w,i	SEG =SS	AAA
4	INC AX	INC CX	INC DX	INC BX	INC SP	INC BP	INC SI	INC DI
5	PUSH AX	PUSH CX	PUSH DX	PUSH BX	PUSH SP	PUSH BP	PUSH SI	PUSH DI
6								
7	JO	JNO	JB/ JNAE	JNB/ JAE	JE/ JZ	JNE/ JNZ	JBE/ JNA	JNBE/ JA
8	Immed b,r/m	Immed w,r/m	Immed b,r/m	Immed is,r/m	TEST b,r/m	TEST w,r/m	XCHG b,r/m	XCHG w,r/m
9	NOP	XCHG CX	XCHG DX	XCHG BX	XCHG SP	XCHG BP	XCHG SI	XCHG DI
A	MOV m AL	MOV m AL	MOV AL m	MOV AL m	MOVS b	MOVS w	CMPS b	CMPS w
B	MOV i AL	MOV i CL	MOV i DL	MOV i BL	MOV i AH	MOV i CH	MOV i DH	MOV i BH
C			RET (i+SP)	RET	LES	LDS	MOV b,i,r/m	MOV w,i,r/m
D	Shift b	Shift w	Shift b,v	Shift w,v	AAM	AAD		XLAT
E	LOOPNZ/ LOOPNE	LOOPZ/ LOOPE	LOOP	JCXZ	IN b	IN w	OUT b	OUT w
F	LOCK		REP	REP z	HLT	CMC	Grp 1 b,r/m	Grp 1 w,r/m

b = byte operation
d = direct
f = from CPU reg
i = immediate
ia = immed. to accum.
id = indirect
is = immed. byte, sign ext.
l = long ie. intersegment

m = memory
r/m = EA is second byte
si = short intrasegment
sr = segment register
t = to CPU reg
v = variable
w = word operation
z = zero

8088 Instruction Set Matrix

HI \ LO	8	9	A	B	C	D	E	F
0	OR b,f,r/m	w,f,r/m	OR b,t,r/m	OR w,t,r/m	OR b,i	OR w,i	PUSH CS	
1	SBB b,f,r/m	SBB w,f,r/m	SBB b,t,r/m	SBB w,t,r/m	SBB b,i	SBB w,i	PUSH DS	POP DS
2	SUB b,f,r/m	SUB w,f,r/m	SUB b,t,r/m	SUB w,t,r/m	SUB b,i	SUB w,i	SEG= CS	DAS
3	CMP b,f,r/m	CMP w,f,r/m	CMP b,t,r/m	CMP w,t,r/m	CMP b,i	CMP w,i	SEG= CS	AAS
4	DEC AX	DEC CX	DEC DX	DEC BX	DEC SP	DEC BP	DEC SI	DEC DI
5	POP AX	POP CX	POP DX	POP BX	POP SP	POP BP	POP SI	POP DI
6								
7	JS	JNS	JP/ JPE	JNP/ JPO	JL/ JNGE	JNL/ JGE	JLE/ JNG	JNLE/ JG
8	MOV b,f,r/m	MOV w,f,r/m	MOV b,t,r/m	MOV w,t,r/m	MOV sr,t,r/m	LEA	MOV sr,f,r/m	POP r/m
9	CBW	CWD	CALL l,d	WAIT	PUSHF	POPF	SAHF	LAHF
A	TEST b,i	TEST w,i	STOS b	STOS w	LODS b	LODS w	SCAS b	SCAS w
B	MOV i AX	MOV i CX	MOV i DX	MOV i BX	MOV i SP	MOV i BP	MOV i SI	MOV i DI
C			RET l,(i+SP)	RET l	INT Type 3	INT (Any)	INTO	IRET
D	ESC 0	ESC 1	ESC 2	ESC 3	ESC 4	ESC 5	ESC 6	ESC 7
E	CALL d	JMP d	JMP l,d	JMP si,d	IN v,b	IN v,w	OUT v,b	OUT v,w
F	CLC	STC	CLI	STI	CLD	STD	Grp 2 b,r/m	Grp 2 w,r/m

where:

mod□r/m	000	001	010	011	100	101	110	111
Immed	ADD	OR	ADC	SBB	AND	SUB	XOR	CMP
Shift	ROL	ROR	RCL	RCR	SHL/SAL	SHR	—	SAR
Grp 1	TEST	—	NOT	NEG	MUL	IMUL	DIV	IDIV
Grp 2	INC	DEC	CALL id	CALL l,id	JMP id	JMP l,id	PUSH	—

Appendix B

Instruction Set Index

Mnemonic	Page
AAA	B-7
AAD	B-9
AAM	B-8
AAS	B-8
ADC	B-7
ADD	B-7
AND	B-9
CALL	B-11
CBW	B-9
CLC	B-15
CLD	B-15
CLI	B-15
CMC	B-15
CMP	B-8
CMPS	B-10
CWD	B-9
DAA	B-7
DAS	B-8
DEC	B-8
DIV	B-8
ESC	B-15
HLT	B-15
IDIV	B-9
IMUL	B-8
IN	B-6
INC	B-7
INT	B-14
INTO	B-14
IRET	B-14
JA	B-13
JAE	B-13
JB	B-12
JBE	B-12
JCXZ	B-13
JE	B-12
JG	B-13
JGE	B-12
JL	B-12
JLE	B-12
JMP	B-11
JNA	B-12
JNAE	B-12
JNB	B-13
JNBE	B-13
JNE	B-12
JNG	B-12
JNGE	B-12
JNL	B-12
JNLE	B-13
JNO	B-13
JNP	B-13
JNS	B-13
JNZ	B-12
JO	B-12
JP	B-12
JPE	B-12
JPO	B-13
JS	B-12
JZ	B-12
LAHF	B-6
LDS	B-6
LEA	B-6
LES	B-6
LOCK	B-15
LODS	B-11
LOOP	B-13
LOOPE	B-13
LOOPNE	B-13
LOOPNZ	B-13
LOOPZ	B-13
MOV	B-5
MOVS	B-10
MUL	B-8
NEG	B-8
NOP	B-15
NOT	B-9
OR	B-10
OUT	B-6
POP	B-5
POPF	B-6
PUSH	B-5
PUSHF	B-6
RCL	B-9
RCR	B-9
REP	B-10
RET	B-12
ROL	B-9
ROR	B-9
SAHF	B-6
SAL	B-9
SAR	B-9
SBB	B-8
SCAS	B-10
SHL	B-9
SHR	B-9
STC	B-15
STD	B-15
STI	B-15
STOS	B-11
SUB	B-7
TEST	B-10
WAIT	B-15
XCHG	B-6
XLAT	B-6
XOR	B-10

APPENDIX C: OF CHARACTERS, KEYSTROKES, AND COLOR

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
00	0	Blank (Null)	Ctrl 2		Black	Black	Non-Display
01	1	☺	Ctrl A		Black	Blue	Underline
02	2	☻	Ctrl B		Black	Green	Normal
03	3	♥	Ctrl C		Black	Cyan	Normal
04	4	♦	Ctrl D		Black	Red	Normal
05	5	♣	Ctrl E		Black	Magenta	Normal
06	6	♠	Ctrl F		Black	Brown	Normal
07	7	•	Ctrl G		Black	Light Grey	Normal
08	8	◘	Ctrl H, Backspace, Shift Backspace		Black	Dark Grey	Non-Display
09	9	○	Ctrl I		Black	Light Blue	High Intensity Underline
0A	10	◙	Ctrl J, Ctrl ↵		Black	Light Green	High Intensity
0B	11	♂	Ctrl K		Black	Light Green	High Intensity
0C	12	♀	Ctrl L,		Black	Light Red	High Intensity
0D	13	♪	Ctrl M, ↵, Shift ↵		Black	Light Magenta	High Intensity
0E	14	♫	Ctrl N		Black	Yellow	High Intensity
0F	15	☼	Ctrl O		Black	White	High Intensity
10	16	►	Ctrl P		Blue	Black	Normal
11	17	◄	Ctrl Q		Blue	Blue	Underline
12	18	↕	Ctrl R		Blue	Green	Normal
13	19	‼	Ctrl S		Blue	Cyan	Normal
14	20	¶	Ctrl T		Blue	Red	Normal
15	21	§	Ctrl U			Magenta	Normal
16	22	▬	Ctrl V		Blue	Brown	Normal
17	23	↨	Ctrl W		Blue	Light Grey	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
18	24	↑	Ctrl X		Blue	Dark Grey	High Intensity
19	25	↓	Ctrl Y		Blue	Light Blue	High Intensity Underline
1A	26	→	Ctrl Z		Blue	Light Green	High Intensity
1B	27	←	Ctrl [, Esc, Shift Esc, Ctrl Esc		Blue	Light Cyan	High Intensity
1C	28	∟	Ctrl \		Blue	Light Red	High Intensity
1D	29	↔	Ctrl]		Blue	Light Magenta	High Intensity
1E	30	▲	Ctrl 6		Blue	Yellow	High Intensity
1F	31	▼	Ctrl –		Blue	White	High Intensity
20	32	Blank Space	Space Bar, Shift, Space, Ctrl Space, Alt Space		Green	Black	Normal
21	33	!	!	Shift	Green	Blue	Underline
22	34	"	"	Shift	Green	Green	Normal
23	35	#	#	Shift	Green	Cyan	Normal
24	36	$	$	Shift	Green	Red	Normal
25	37	%	%	Shift	Green	Magenta	Normal
26	38	&	&	Shift	Green	Brown	Normal
27	39	'	'		Green	Light Grey	Normal
28	40	(	(	Shift	Green	Dark Grey	High Intensity
29	41	)	)	Shift	Green	Light Blue	High Intensity Underline
2A	42	*	*	Note 1	Green	Light Green	High Intensity
28	43	+	+	Shift	Green	Light Cyan	High Intensity
2C	44	,	,		Green	Light Red	High Intensity
2D	45	—	—		Green	Light Magenta	High Intensity
2E	46	.	.	Note 2	Green	Yellow	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
2F	47	/	/		Green	White	High Intensity
30	48	0	0	Note 3	Cyan	Black	Normal
31	49	1	1	Note 3	Cyan	Blue	Underline
32	50	2	2	Note 3	Cyan	Green	Normal
33	51	3	3	Note 3	Cyan	Cyan	Normal
34	52	4	4	Note 3	Cyan	Red	Normal
35	53	5	5	Note 3	Cyan	Magenta	Normal
36	54	6	6	Note 3	Cyan	Brown	Normal
37	55	7	7	Note 3	Cyan	Light Grey	Normal
38	56	8	8	Note 3	Cyan	Dark Grey	High Intensity
39	57	9	9	Note 3	Cyan	Light Blue	High Intensity Underline
3A	58	:	:	Shift	Cyan	Light Green	High Intensity
3B	59	;	;		Cyan	Light Cyan	High Intensity
3C	60	<	<	Shift	Cyan	Light Red	High Intensity
3D	61	=	=		Cyan	Light Magenta	High Intensity
3E	62	>	>	Shift	Cyan	Yellow	High Intensity
3F	63	?	?	Shift	Cyan	White	High Intensity
40	64	@	@	Shift	Red	Black	Normal
41	65	A	A	Note 4	Red	Blue	Underline
42	66	B	B	Note 4	Red	Green	Normal
43	67	C	C	Note 4	Red	Cyan	Normal
44	68	D	D	Note 4	Red	Red	Normal
45	69	E	E	Note 4	Red	Magenta	Normal
46	70	F	F	Note 4	Red	Brown	Normal
47	71	G	G	Note 4	Red	Light Grey	Normal
48	72	H	H	Note 4	Red	Dark Grey	High Intensity
49	73	I	I	Note 4	Red	Light Blue	High Intensity Underline
4A	74	J	J	Note 4	Red	Light Green	High Intensity

Appendix C

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
4B	75	K	K	Note 4	Red	Light Cyan	High Intensity
4C	76	L	L	Note 4	Red	Light Red	High Intensity
4D	77	M	M	Note 4	Red	Light Magenta	High Intensity
4E	78	N	N	Note 4	Red	Yellow	High Intensity
4F	79	O	O	Note 4	Red	White	High Intensity
50	80	P	P	Note 4	Magenta	Black	Normal
51	81	Q	Q	Note 4	Magenta	Blue	Underline
52	82	R	R	Note 4	Magenta	Green	Normal
53	83	S	S	Note 4	Magenta	Cyan	Normal
54	84	T	T	Note 4	Magenta	Red	Normal
55	85	U	U	Note 4	Magenta	Magenta	Normal
56	86	V	V	Note 4	Magenta	Brown	Normal
57	87	W	W	Note 4	Magenta	Light Grey	Normal
58	88	X	X	Note 4	Magenta	Dark Grey	High Intensity
59	89	Y	Y	Note 4	Magenta	Light Blue	High Intensity Underline
5A	90	Z	Z	Note 4	Magenta	Light Green	High Intensity
5B	91	[	[		Magenta	Light Cyan	High Intensity
5C	92	\	\		Magenta	Light Red	High Intensity
5D	93	]	]		Magenta	Light Magenta	High Intensity
5E	94	^	^	Shift	Magenta	Yellow	High Intensity
5F	95	—	—	Shift	Magenta	White	High Intensity
60	96	'	'		Yellow	Black	Normal
61	97	a	a	Note 5	Yellow	Blue	Underline
62	98	b	b	Note 5	Yellow	Green	Normal
63	99	c	c	Note 5	Yellow	Cyan	Normal
64	100	d	d	Note 5	Yellow	Red	Normal
65	101	e	e	Note 5	Yellow	Magenta	Normal
66	102	f	f	Note 5	Yellow	Brown	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
67	103	g	g	Note 5	Yellow	Light Grey	Normal
68	104	h	h	Note 5	Yellow	Dark Grey	High Intensity
69	105	i	i	Note 5	Yellow	Light Blue	High Intensity Underline
6A	106	j	j	Note 5	Yellow	Light Green	High Intensity
6B	107	k	k	Note 5	Yellow	Light Cyan	High Intensity
6C	108	l	l	Note 5	Yellow	Light Red	High Intensity
6D	109	m	m	Note 5	Yellow	Light Magenta	High Intensity
6E	110	n	n	Note 5	Yellow	Yellow	High Intensity
6F	111	o	o	Note 5	Yellow	White	High Intensity
70	112	p	p	Note 5	White	Black	Reverse Video
71	113	q	q	Note 5	White	Blue	Underline
72	114	r	r	Note 5	White	Green	Normal
73	115	s	s	Note 5	White	Cyan	Normal
74	116	f	f	Note 5	White	Red	Normal
75	117	u	u	Note 5	White	Magenta	Normal
76	118	v	v	Note 5	White	Brown	Normal
77	119	w	w	Note 5	White	Light Grey	Normal
78	120	x	x	Note 5	White	Dark Grey	Reverse Video
79	121	y	y	Note 5	White	Light Blue	High Intensity Underline
7A	122	z	z	Note 5	White	Light Green	High Intensity
7B	123	{	{	Shift	White	Light Cyan	High Intensity
7C	124	¦	¦	Shift	White	Light Red	High Intensity
7D	125	}	}	Shift	White	Light Magenta	High Intensity
7E	126	~	~	Shift	White	Yellow	High Intensity
7F	127	Δ	Ctrl ←		White	White	High Intensity

Appendix C

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
* * * * 80 to FF Hex are Flashing in both Color & IBM Monochrome * * * *							
80	128	Ç	Alt 128	Note 6	Black	Black	Non-Display
81	129	ü	Alt 129	Note 6	Black	Blue	Underline
82	130	é	Alt 130	Note 6	Black	Green	Normal
83	131	â	Alt 131	Note 6	Black	Cyan	Normal
84	132	ä	Alt 132	Note 6	Black	Red	Normal
85	133	à	Alt 133	Note 6	Black	Magenta	Normal
86	134	å	Alt 134	Note 6	Black	Brown	Normal
87	135	ç	Alt 135	Note 6	Black	Light Grey	Normal
88	136	ê	Alt 136	Note 6	Black	Dark Grey	Non-Display
89	137	ë	Alt 137	Note 6	Black	Light Blue	High Intensity Underline
8A	138	è	Alt 138	Note 6	Black	Light Green	High Intensity
8B	139	ï	Alt 139	Note 6	Black	Light Cyan	High Intensity
8C	140	î	Alt 140	Note 6	Black	Light Red	High Intensity
8D	141	ì	Alt 141	Note 6	Black	Light Magenta	High Intensity
8E	142	Ä	Alt 142	Note 6	Black	Yellow	High Intensity
8F	143	Å	Alt 143	Note 6	Black	White	High Intensity
90	144	É	Alt 144	Note 6	Blue	Black	Normal
91	145	æ	Alt 145	Note 6	Blue	Blue	Underline
92	146	Æ	Alt 146	Note 6	Blue	Green	Normal
93	147	ô	Alt 147	Note 6	Blue	Cyan	Normal
94	148	ö	Alt 148	Note 6	Blue	Red	Normal
95	149	ò	Alt 149	Note 6	Blue	Magenta	Normal
96	150	û	Alt 150	Note 6	Blue	Brown	Normal
97	151	ù	Alt 151	Note 6	Blue	Light Grey	Normal
98	152	ÿ	Alt 152	Note 6	Blue	Dark Grey	High Intensity
99	153	Ö	Alt 153	Note 6	Blue	Light Blue	High Intensity Underline
9A	154	Ü	Alt 154	Note 6	Blue	Light Green	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
9B	155	¢	Alt 155	Note 6	Blue	Light Cyan	High Intensity
9C	156	£	Alt 156	Note 6	Blue	Light Red	High Intensity
9D	157	¥	Alt 157	Note 6	Blue	Light Magenta	High Intensity
9E	158	Pt	Alt 158	Note 6	Blue	Yellow	High Intensity
9F	159	ƒ	Alt 159	Note 6	Blue	White	High Intensity
A0	160	á	Alt 160	Note 6	Green	Black	Normal
A1	161	í	Alt 161	Note 6	Green	Blue	Underline
A2	162	ó	Alt 162	Note 6	Green	Green	Normal
A3	163	ú	Alt 163	Note 6	Green	Cyan	Normal
A4	164	ñ	Alt 164	Note 6	Green	Red	Normal
A5	165	Ñ	Alt 165	Note 6	Green	Magenta	Normal
A6	166	ª	Alt 166	Note 6	Green	Brown	Normal
A7	167	º	Alt 167	Note 6	Green	Light Grey	Normal
A8	168	¿	Alt 168	Note 6	Green	Dark Grey	High Intensity
A9	169	⌐	Alt 169	Note 6	Green	Light Blue	High Intensity Underline
AA	170	¬	Alt 170	Note 6	Green	Light Green	High Intensity
AB	171	½	Alt 171	Note 6	Green	Light Cyan	High Intensity
AC	172	¼	Alt 172	Note 6	Green	Light Red	High Intensity
AD	173	¡	Alt 173	Note 6	Green	Light Magenta	High Intensity
AE	174	«	Alt 174	Note 6	Green	Yellow	High Intensity
AF	175	»	Alt 175	Note 6	Green	White	High Intensity
B0	176	░	Alt 176	Note 6	Cyan	Black	Normal
B1	177	▒	Alt 177	Note 6	Cyan	Blue	Underline
B2	178	▓	Alt 178	Note 6	Cyan	Green	Normal
B3	179	│	Alt 179	Note 6	Cyan	Cyan	Normal
B4	180	┤	Alt 180	Note 6	Cyan	Red	Normal
B5	181	╡	Alt 181	Note 6	Cyan	Magenta	Normal
B6	182	╢	Alt 182	Note 6	Cyan	Brown	Normal

Appendix C

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
B7	183	╖	Alt 183	Note 6	Cyan	Light Grey	Normal
B8	184	╕	Alt 184	Note 6	Cyan	Dark Grey	High Intensity
B9	185	╣	Alt 185	Note 6	Cyan	Light Blue	High Intensity Underline
BA	1B6	║	Alt 186	Note 6	Cyan	Light Green	High Intensity
BB	187	╗	Alt 187	Note 6	Cyan	Light Cyan	High Intensity
BC	1BB	╝	Alt 1B8	Note 6	Cyan	Light Red	High Intensity
BD	189	╜	Alt 189	Note 6	Cyan	Light Magenta	High Intensity
BE	190	╛	Alt 190	Note 6	Cyan	Yellow	High Intensity
BF	191	┐	Alt 191	Note 6	Cyan	White	High Intensity
C0	192	└	Alt 192	Note 6	Red	Black	Normal
C1	193	┴	Alt 193	Note 6	Red	Blue	Underline
C2	194	┬	Alt 194	Note 6	Red	Green	Normal
C3	195	├	Alt 195	Note 6	Red	Cyan	Normal
C4	196	─	Alt 196	Note 6	Red	Red	Normal
C5	197	┼	Alt 197	Note 6	Red	Magenta	Normal
C6	198	╞	Alt 198	Note 6	Red	Brown	Normal
C7	199	╟	Alt 199	Note 6	Red	Light Grey	Normal
C8	200	╚	Alt 200	Note 6	Red	Dark Grey	High Intensity
C9	201	╔	Alt 201	Note 6	Red	Light Blue	High Intensity Underline
CA	202	╩	Alt 202	Note 6	Red	Light Green	High Intensity
CB	203	╦	Alt 203	Note 6	Red	Light Cyan	High Intensity
CC	204	╠	Alt 204	Note 6	Red	Light Red	High Intensity
CD	205	═	Alt 205	Note 6	Red	Light Magenta	High Intensity
CE	206	╬	Alt 206	Note 6	Red	Yellow	High Intensity
CF	207	╧	Alt 207	Note 6	Red	White	High Intensity
D0	208	╨	Alt 208	Note 6	Magenta	Black	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
D1	209	╤	Alt 209	Note 6	Magenta	Blue	Underline
D2	210	╥	Alt 210	Note 6	Magenta	Green	Normal
D3	211	╙	Alt 211	Note 6	Magenta	Cyan	Normal
D4	212	╘	Alt 212	Note 6	Magenta	Red	Normal
D5	213	╒	Alt 213	Note 6	Magenta	Magenta	Normal
D6	214	╓	Alt 214	Note 6	Magenta	Brown	Normal
D7	215	╫	Alt 215	Note 6	Magenta	Light Grey	Normal
D8	216	╪	Alt 216	Note 6	Magenta	Dark Grey	High Intensity
D9	217	┘	Alt 217	Note 6	Magenta	Light Blue	High Intensity Underline
DA	218	┌	Alt 218	Note 6	Magenta	Light Green	High Intensity
DB	219	█	Alt 219	Note 6	Magenta	Light Cyan	High Intensity
DC	220	▄	Alt 220	Note 6	Magenta	Light Red	High Intensity
DD	221	▌	Alt 221	Note 6	Magenta	Light Magenta	High Intensity
DE	222	▐	Alt 222	Note 6	Magenta	Yellow	High Intensity
DF	223	▀	Alt 223	Note 6	Magenta	White	High Intensity
E0	224	α	Alt 224	Note 6	Yellow	Black	Normal
E1	225	β	Alt 225	Note 6	Yellow	Blue	Underline
E2	226	Γ	Alt 226	Note 6	Yellow	Green	Normal
E3	227	π	Alt 227	Note 6	Yellow	Cyan	Normal
E4	228	Σ	Alt 228	Note 6	Yellow	Red	Normal
E5	229	σ	Alt 229	Note 6	Yellow	Magenta	Normal
E6	230	μ	Alt 230	Note 6	Yellow	Brown	Normal
E7	231	τ	Alt 231	Note 6	Yellow	Light Grey	Normal
E8	232	Φ	Alt 232	Note 6	Yellow	Dark Grey	High Intensity
E9	233	θ	Alt 233	Note 6	Yellow	Light Blue	High Intensity Underline
EA	234	Ω	Alt 234	Note 6	Yellow	Light Green	High Intensity
EB	235	δ	Alt 235	Note 6	Yellow	Light Cyan	High Intensity

Appendix C

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
EC	236	∞	Alt 236	Note 6	Yellow	Light Red	High Intensity
ED	237	ϕ	Alt 237	Note 6	Yellow	Light Magenta	High Intensity
EE	238	ϵ	Alt 238	Note 6	Yellow	Yellow	High Intensity
EF	239	∩	Alt 239	Note 6	Yellow	White	High Intensity
F0	240	≡	Alt 240	Note 6	White	Black	Reverse Video
F1	241	±	Alt 241	Note 6	White	Blue	Underline
F2	242	≥	Alt 242	Note 6	White	Green	Normal
F3	243	≤	Alt 243	Note 6	White	Cyan	Normal
F4	244	⌠	Alt 244	Note 6	White	Red	Normal
F5	245	⌡	Alt 245	Note 6	White	Magenta	Normal
F6	246	÷	Alt 246	Note 6	White	Brown	Normal
F7	247	≈	Alt 247	Note 6	White	Light Grey	Normal
F8	248	○	Alt 248	Note 6	White	Dark Grey	Reverse Video
F9	249	●	Alt 249	Note 6	White	Light Blue	High Intensity Underline
FA	250	•	Alt 250	Note 6	White	Light Green	High Intensity
FB	251	√	Alt 251	Note 6	White	Light Cyan	High Intensity
FC	252	η	Alt 252	Note 6	White	Light Red	High Intensity
FD	253	2	Alt 253	Note 6	White	Light Magenta	High Intensity
FE	254	■	Alt 254	Note 6	White	Yellow	High Intensity
FF	255	BLANK	Alt 255	Note 6	White	White	High Intensity

NOTE 1 Asterisk (*) can easily be keyed using two methods: 1) hit the Prt Sc * key or 2) in shift mode hit the * 8 key.

NOTE 2 Period (.) can easily be keyed using two methods: 1) hit the > . key or 2) in shift or Num Lock mode hit the . Del key.

NOTE 3 Numeric characters (0—9) can easily be keyed using two methods: 1) hit the numeric keys on the top row of the typewriter portion of the keyboard or 2) in shift or Num Lock mode hit the numeric keys in the 10—key pad portion of the keyboard.

NOTE 4 Upper case alphabetic characters (A—Z) can easily be keyed in two modes: 1) in shift mode the appropriate alphabetic key or 2) in Caps Lock mode hit the appropriate alphabetic key.

NOTE 5 Lower case alphabetic characters (a—z) can easily be keyed in two modes: 1) in "normal" mode hit the appropriate key or 2) in Caps Lock combined with shift mode hit the appropriate alphabetic key.

NOTE 6 The 3 digits after the Alt key must be typed from the numeric key pad (keys 71—73, 75—77, 79—82). Character codes 000 through 255 can be entered in this fashion. (With Caps Lock activated, Character codes 97 through 122 will display upper case rather than lower case alphabetic characters.)

Character Set (00-7F) Quick Reference

DECIMAL VALUE	➡	0	16	32	48	64	80	96	112
⬇	HEXA-DECIMAL VALUE	0	1	2	3	4	5	6	7
0	0	BLANK (NULL)	►	BLANK (SPACE)	0	@	P	‘	p
1	1	☺	◄	!	1	A	Q	a	q
2	2	☻	↕	"	2	B	R	b	r
3	3	♥	‼	#	3	C	S	c	s
4	4	♦	¶	$	4	D	T	d	t
5	5	♣	§	%	5	E	U	e	u
6	6	♠	▬	&	6	F	V	f	v
7	7	•	↨	'	7	G	W	g	w
8	8	◘	↑	(	8	H	X	h	x
9	9	○	↓	)	9	I	Y	i	y
10	A	◙	→	*	:	J	Z	j	z
11	B	♂	←	+	;	K	[	k	{
12	C	♀	∟	,	<	L	\	l	¦
13	D	♪	↔	—	=	M	]	m	}
14	E	♫	▲	.	>	N	^	n	~
15	F	☼	▼	/	?	O	_	o	△

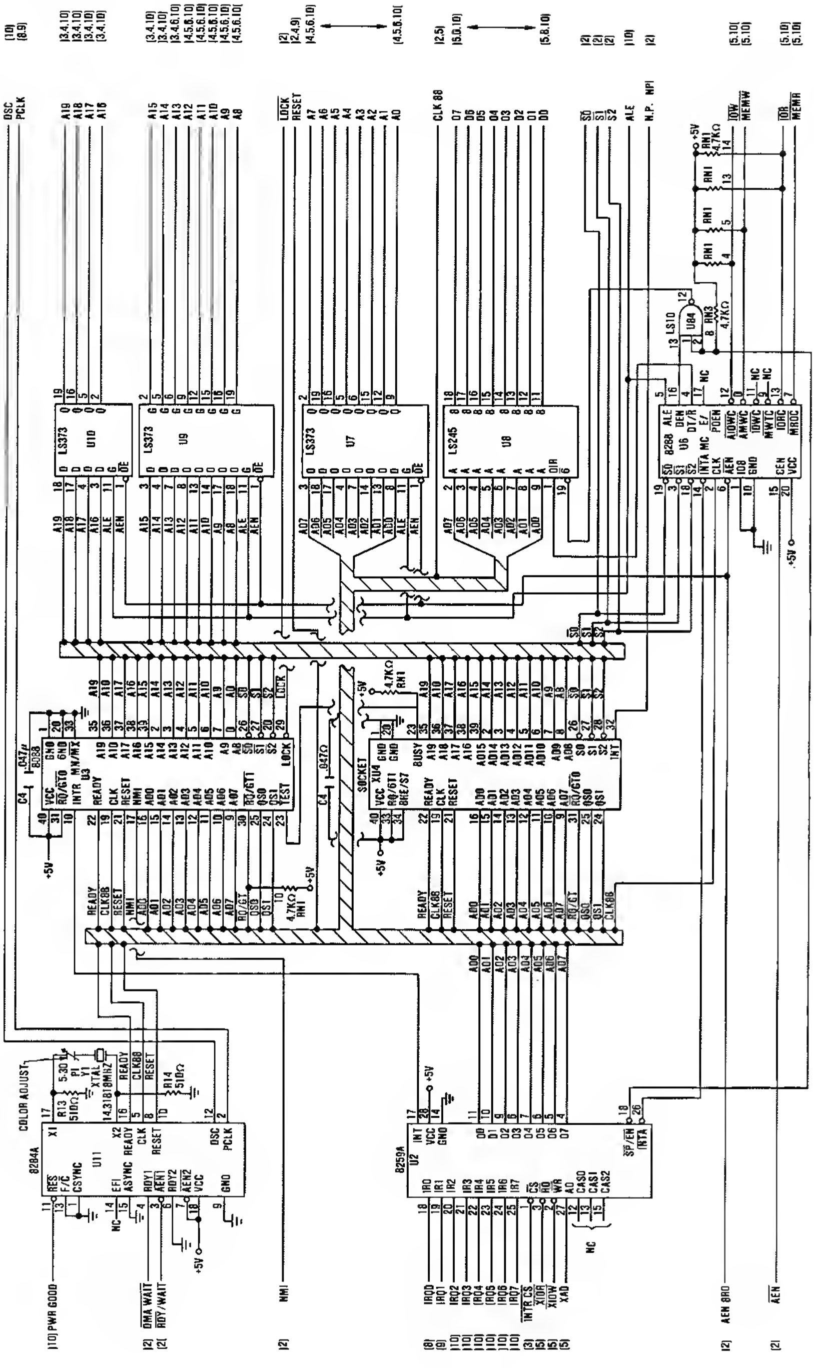
Character Set (80-FF) Quick Reference

DECIMAL VALUE	➡	128	144	160	176	192	208	224	240
⬇	HEXA-DECIMAL VALUE	8	9	A	B	C	D	E	F
0	0	Ç	É	á	░	└	╨	∝	≡
1	1	ü	æ	í	▒	┴	╤	β	±
2	2	é	Æ	ó	▓	┬	╥	Γ	≥
3	3	â	ô	ú	│	├	╙	π	≤
4	4	ä	ö	ñ	┤	─	╘	Σ	⌠
5	5	à	ò	Ñ	╡	┼	╒	σ	⌡
6	6	å	û	ª	╢	╞	╓	µ	÷
7	7	ç	ù	º	╖	╟	╫	τ	≈
8	8	ê	ÿ	¿	╕	╚	╪	Φ	°
9	9	ë	Ö	⌐	╣	╔	┘	Θ	∙
10	A	è	Ü	¬	║	╩	┌	Ω	·
11	B	ï	¢	½	╗	╦	█	δ	√
12	C	î	£	¼	╝	╠	▄	∞	n
13	D	ì	¥	¡	╜	═	▌	φ	2
14	E	Ä	₧	«	╛	╬	▐	∈	■
15	F	Å	ƒ	»	┐	╧	▀	∩	BLANK 'FF'

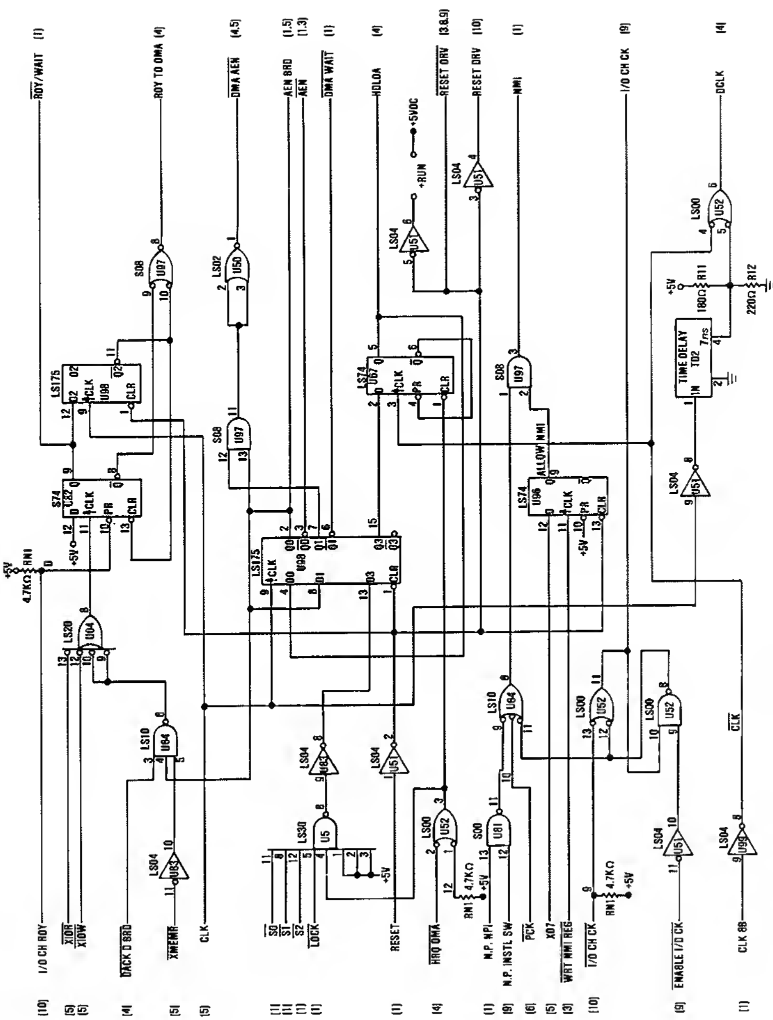
Notes:

APPENDIX D: LOGIC DIAGRAMS

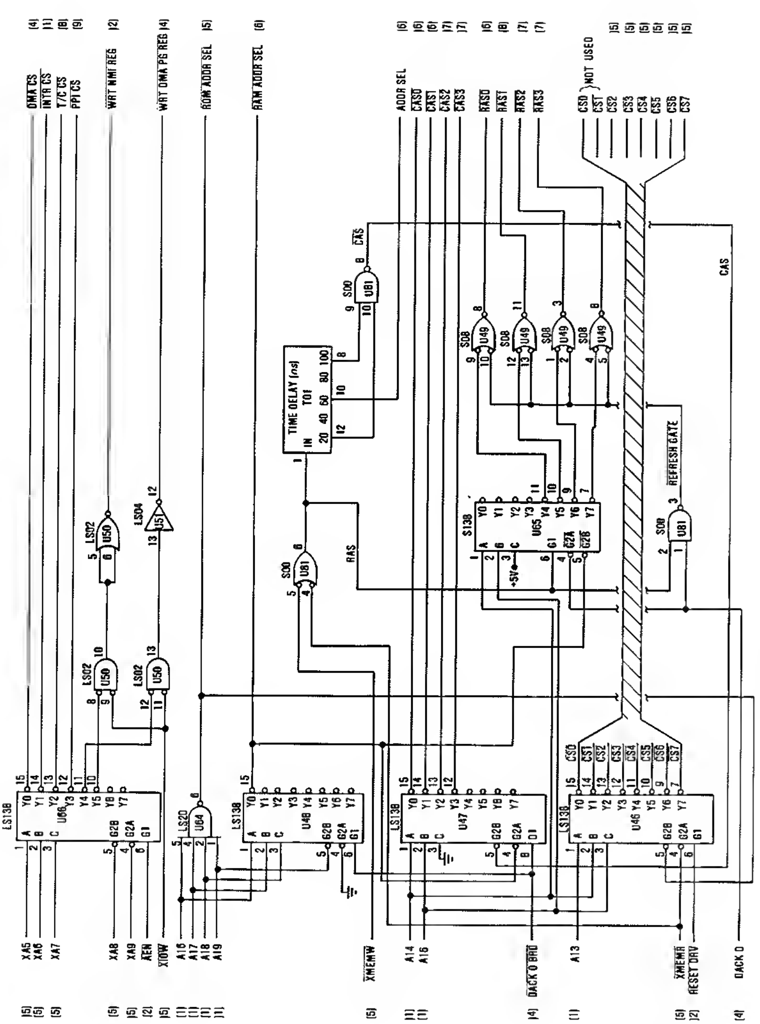
System Board (16/64K) D-2
System Board (64/256K) D-12
Keyboard – Type 1 .... D-22
Keyboard – Type 2 .... D-24
Expansion Board D-25
Extender Card D-26
Receiver Card D-29
Printer D-32
Printer Adapter D-35
Monochrome Display Adapter D-36
Color/Graphics Monitor Adapter D-46
Color Display D-52
Monochrome Display D-54
5–1/4 Inch Diskette Drive Adapter D-55
5–1/4 Inch Diskette Drive – Type 1 .... D-59
5–1/4 Inch Diskette Drive – Type 2 .... D-62
Fixed Disk Drive Adapter D-64
Fixed Disk Drive – Type 1 .... D-70
Fixed Disk Drive – Type 2 .... D-73
32K Memory Expansion Option D-76
64K Memory Expansion Option D-79
64/256K Memory Expansion Option D-82
Game Control Adapter D-86
Prototype Card D-87
Asynchronous Communications Adapter D-88
Binary Synchronous Communications Adapter D-89
SDLC Communications Adapter D-91



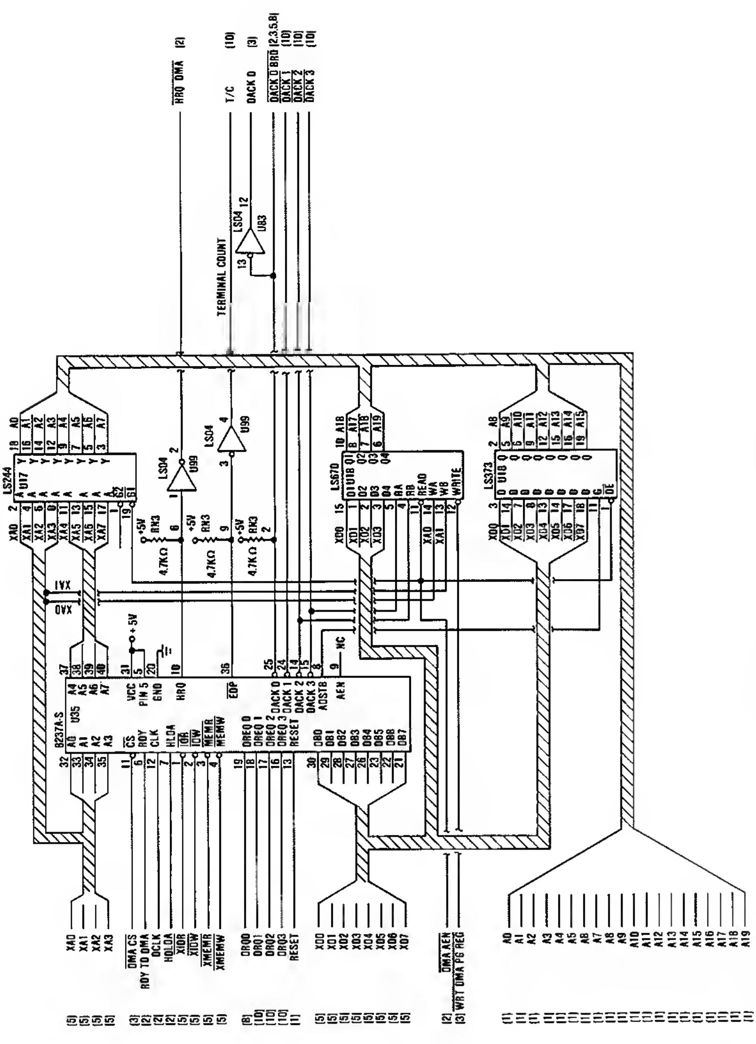
16/64K System Board (Sheet 1 of 10)



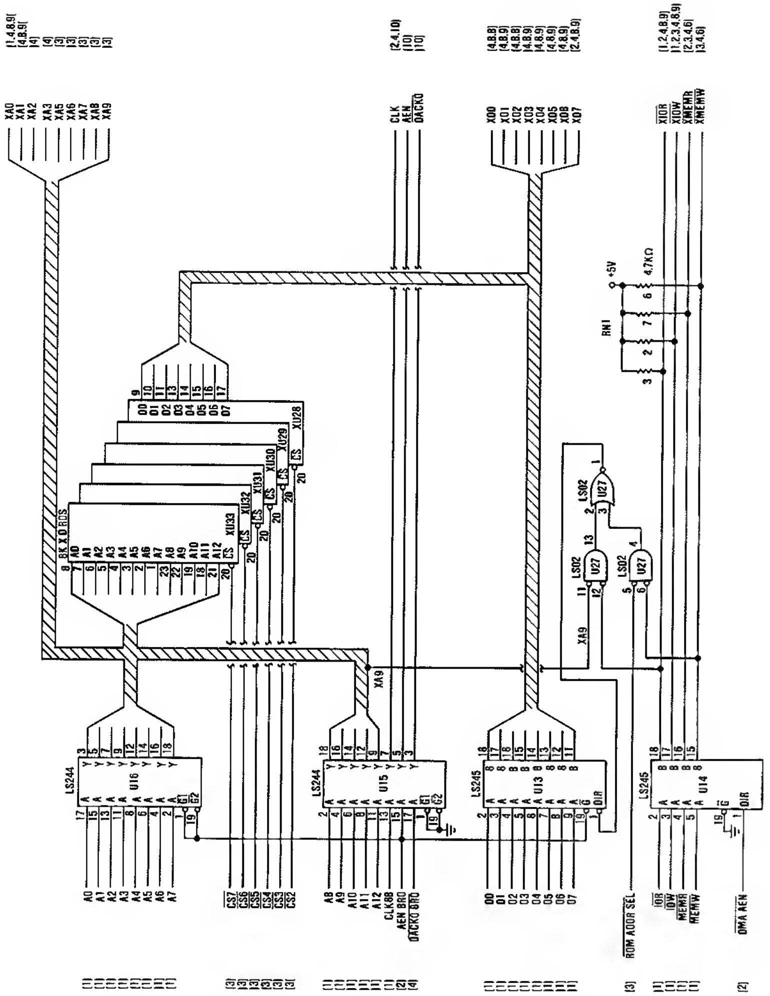
16/64K System Board (Sheet 2 of 10)



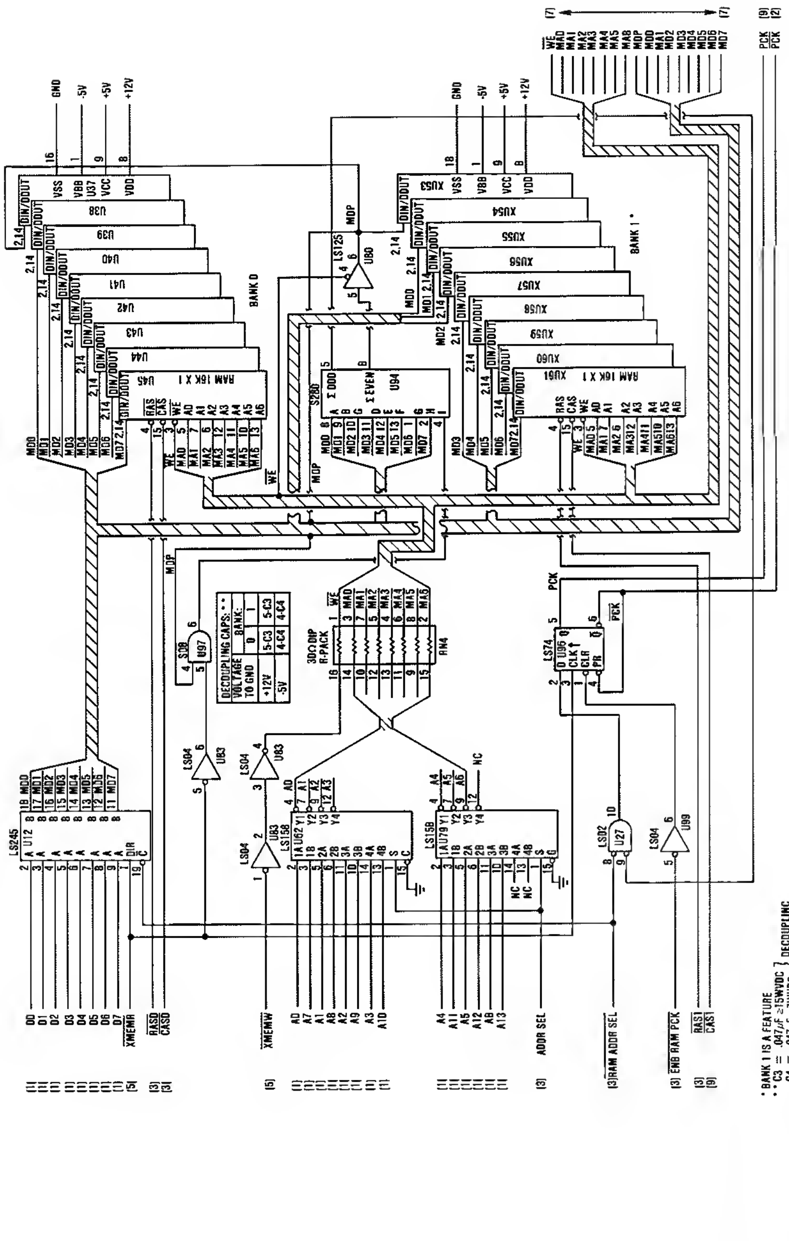
16/64K System Board (Sheet 3 of 10)



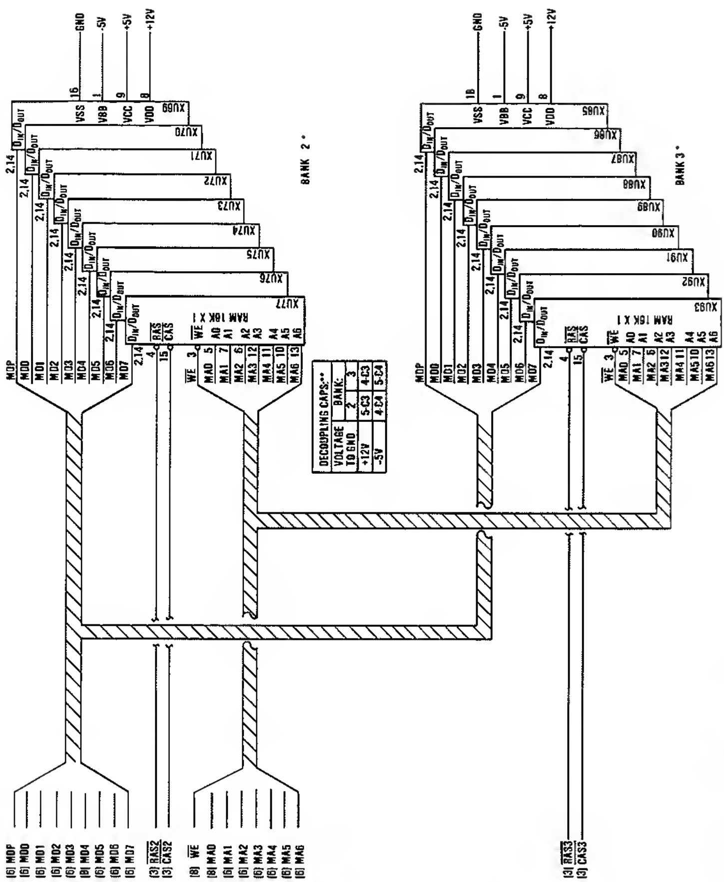
16/64K System Board (Sheet 4 of 10)



16/64K System Board (Sheet 5 of 10)



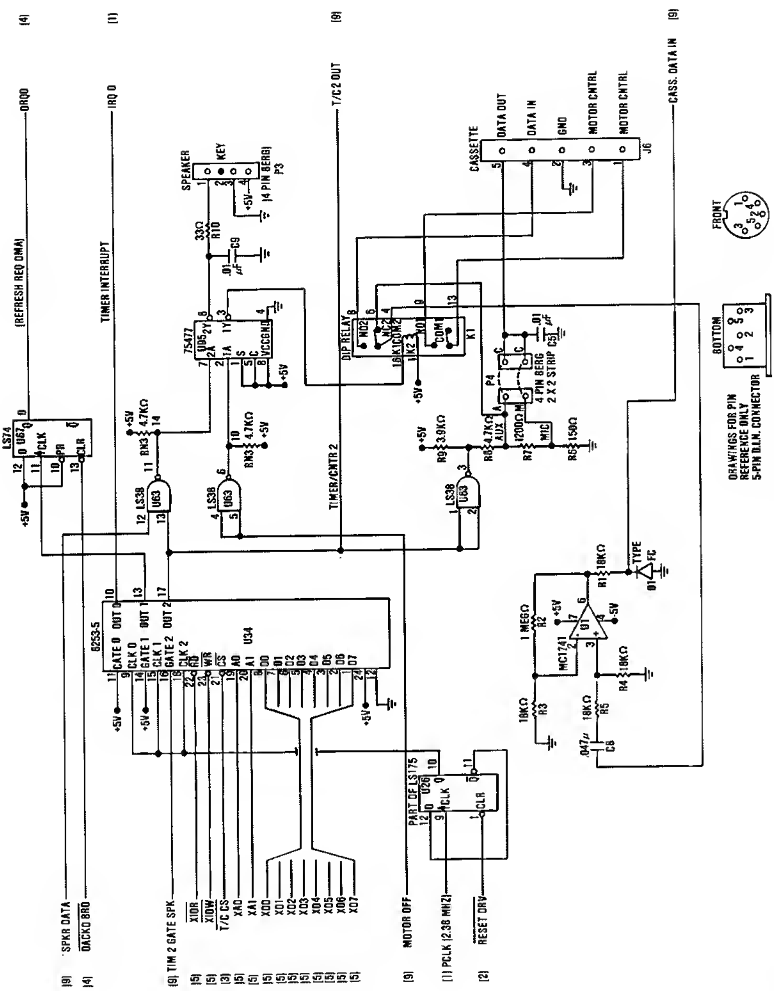
16/64K System Board (Sheet 6 of 10)



* BANKS 2 & 3 ARE FEATURES.

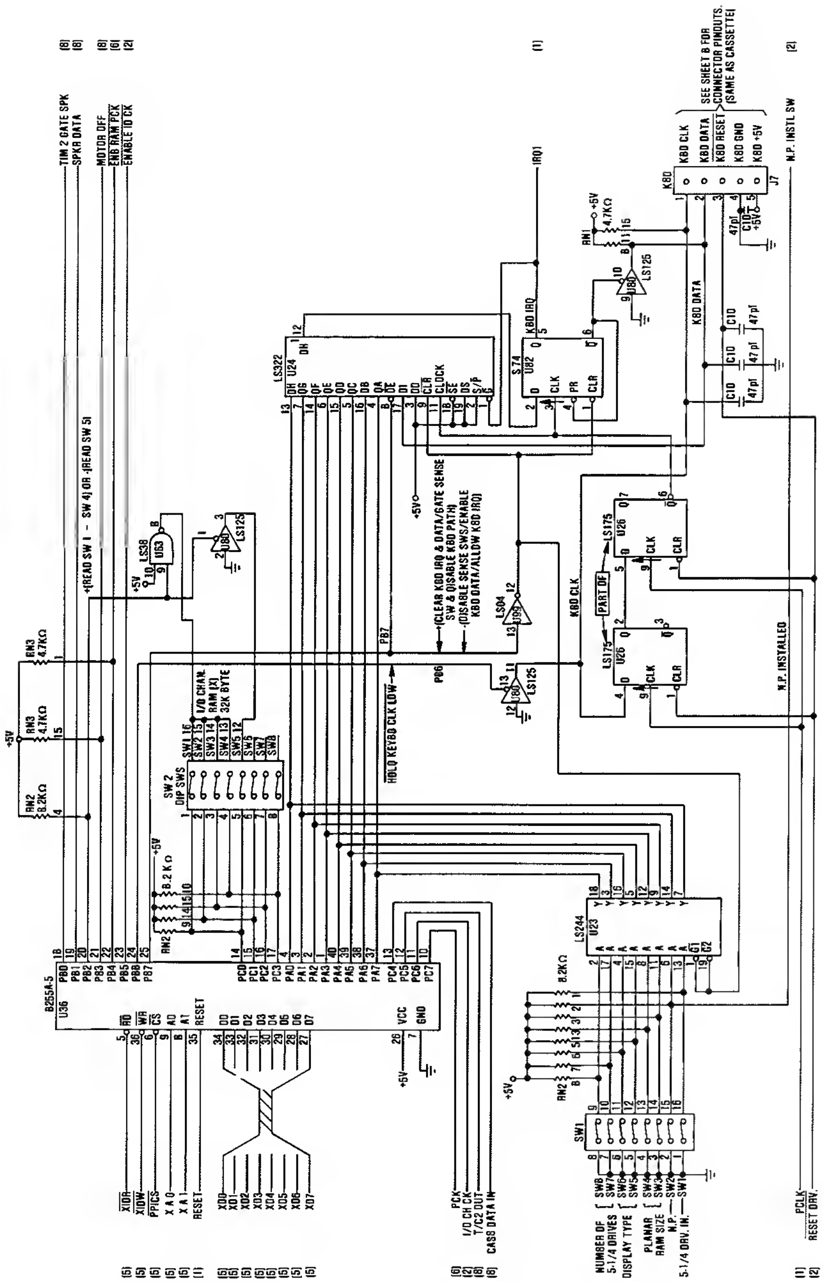
** C3 = .047μF ≥15WVDC } DECOUPLING
C4 = .047μF ≥7WVDC }

16/64K System Board (Sheet 7 of 10)

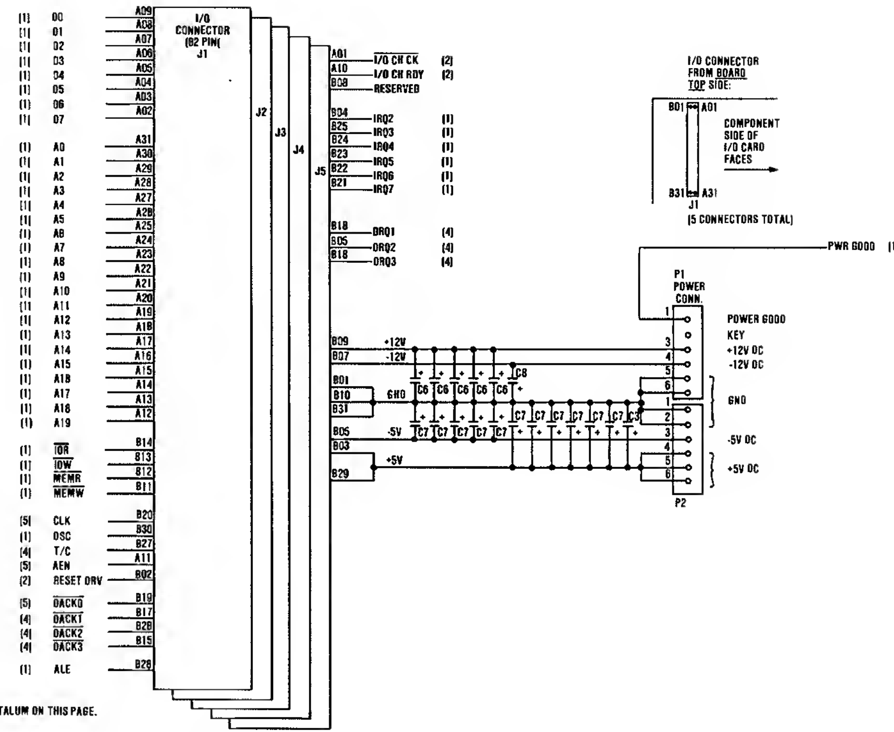


16/64K System Board (Sheet 8 of 10)

Appendix D

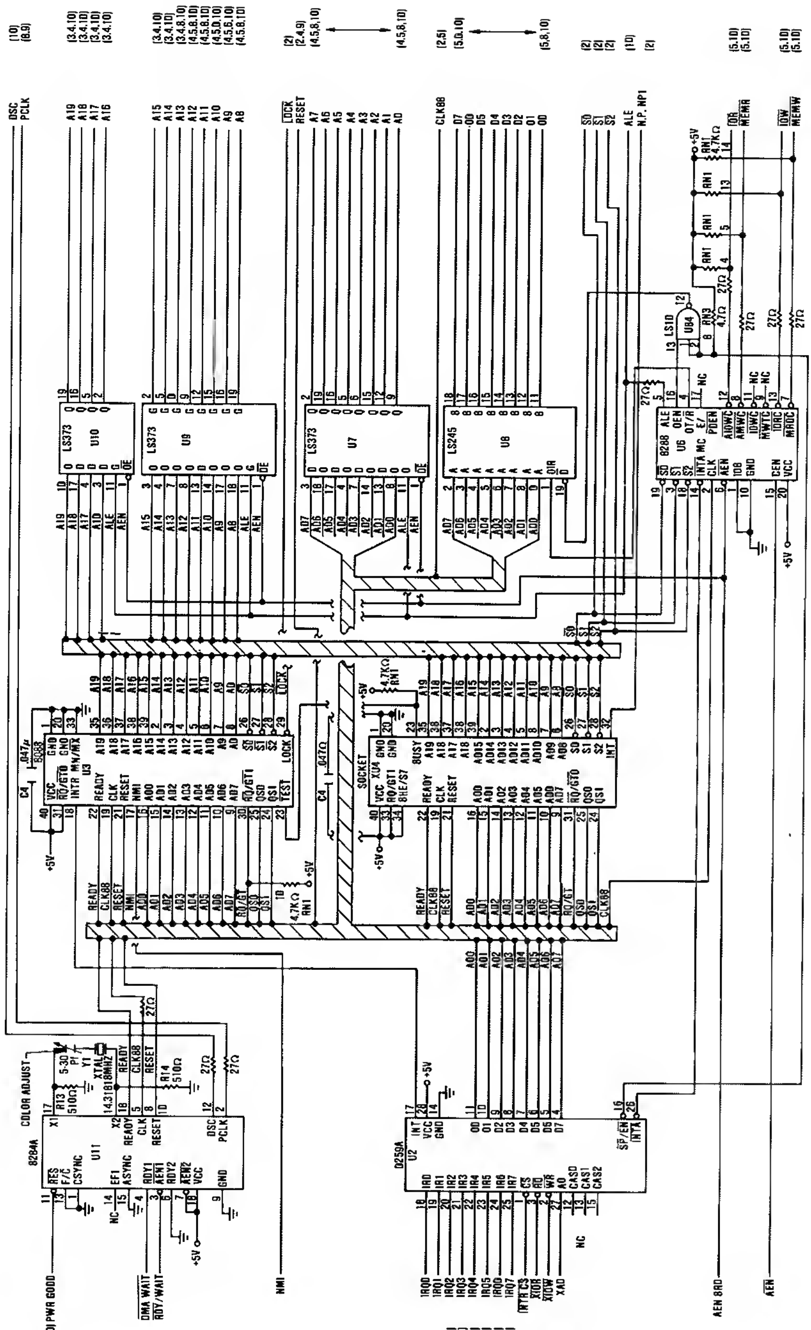


16/64K System Board (Sheet 9 of 10)

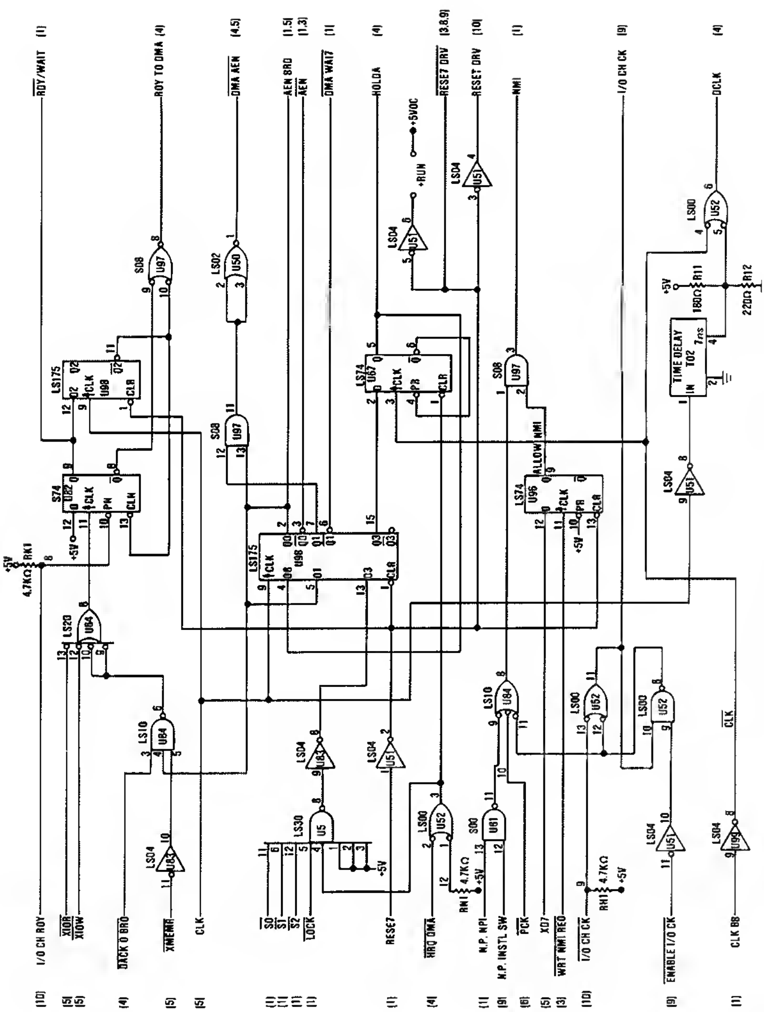


NOTE:
1. ALL CAPS ARE 8.2μF TANTALUM ON THIS PAGE.

16/64K System Board (Sheet 10 of 10)

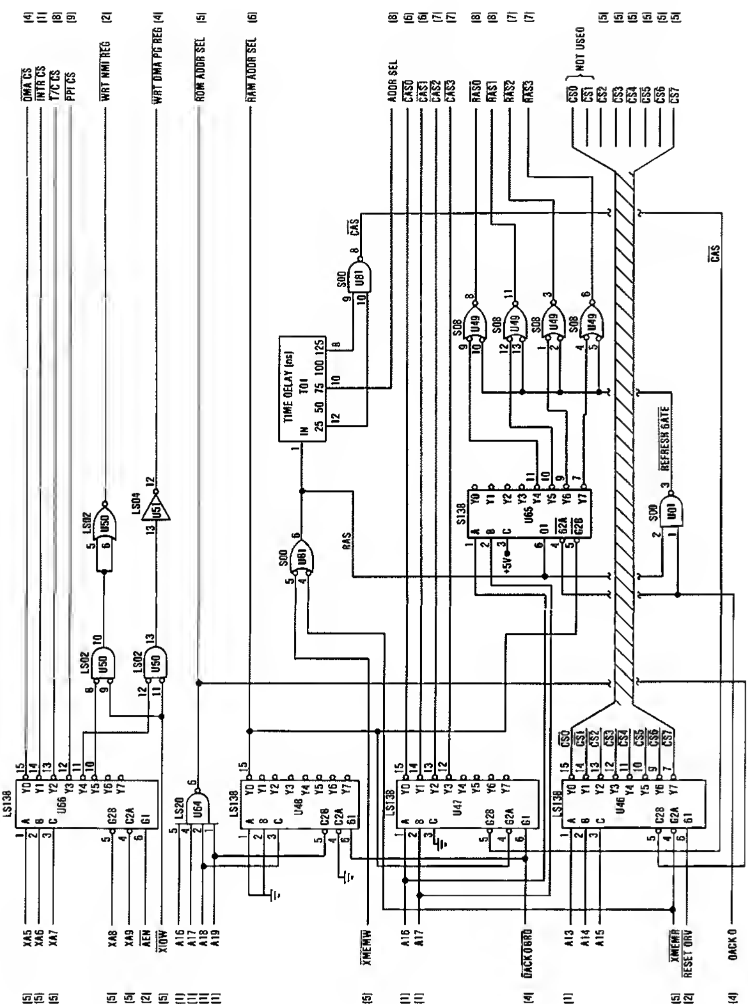


64/256K System Board (Sheet 1 of 10)

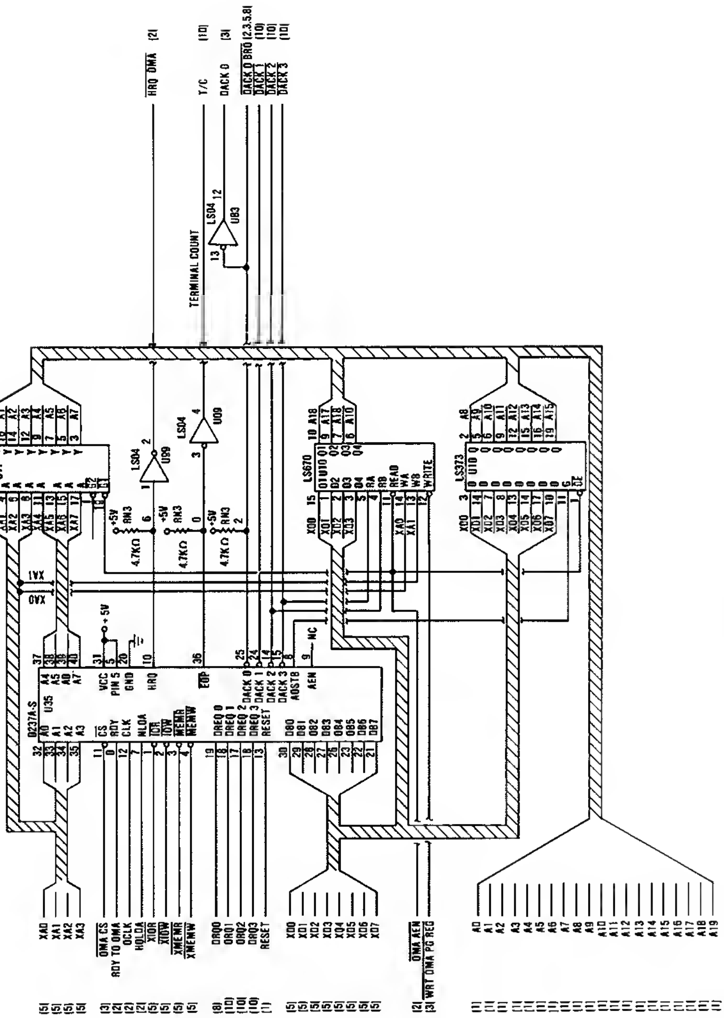


64/256K System Board (Sheet 2 of 10)

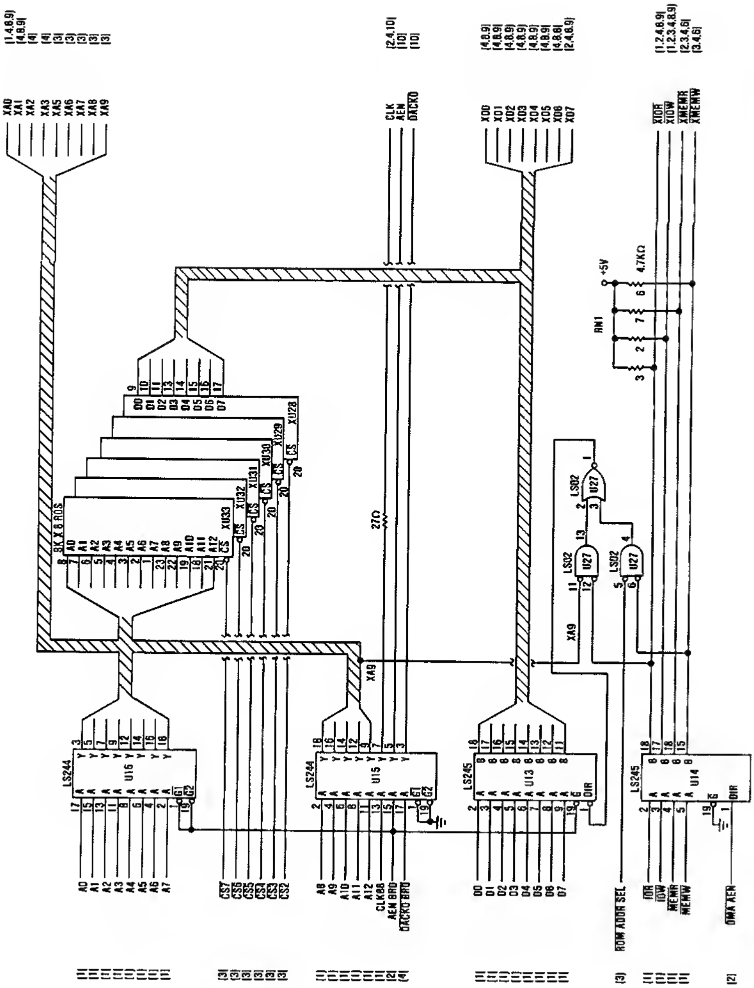
Appendix D



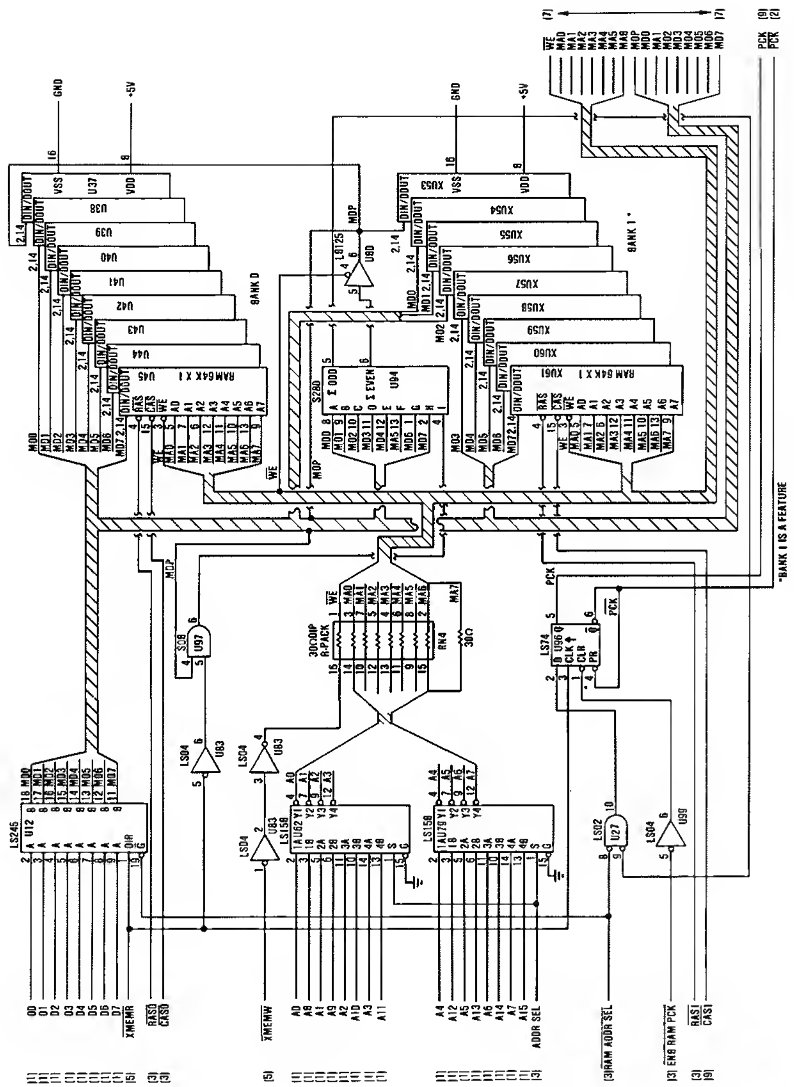
64/256K System Board (Sheet 3 of 10)



64/256K System Board (Sheet 4 of 10)

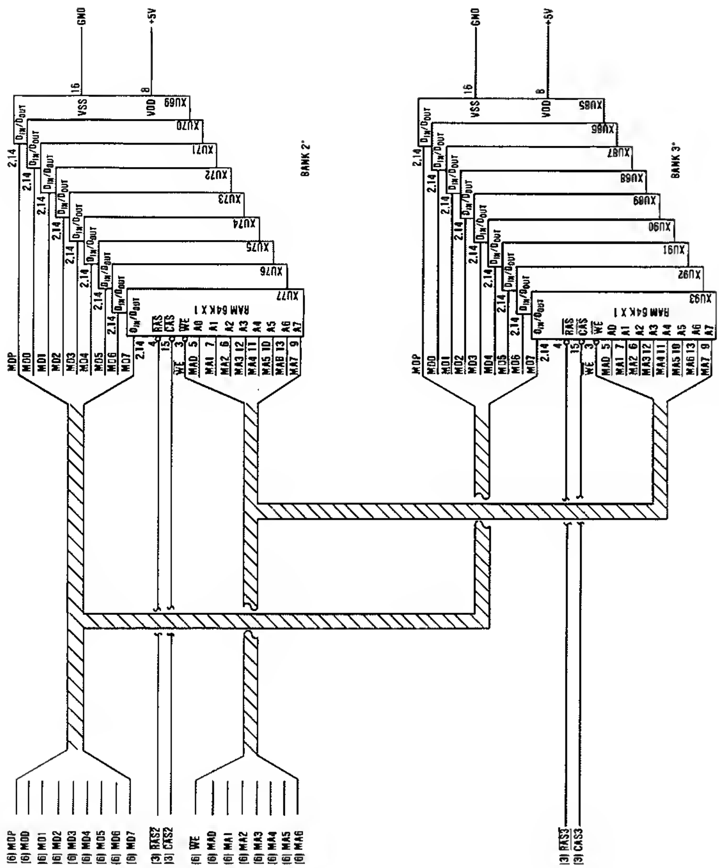


64/256K System Board (Sheet 5 of 10)



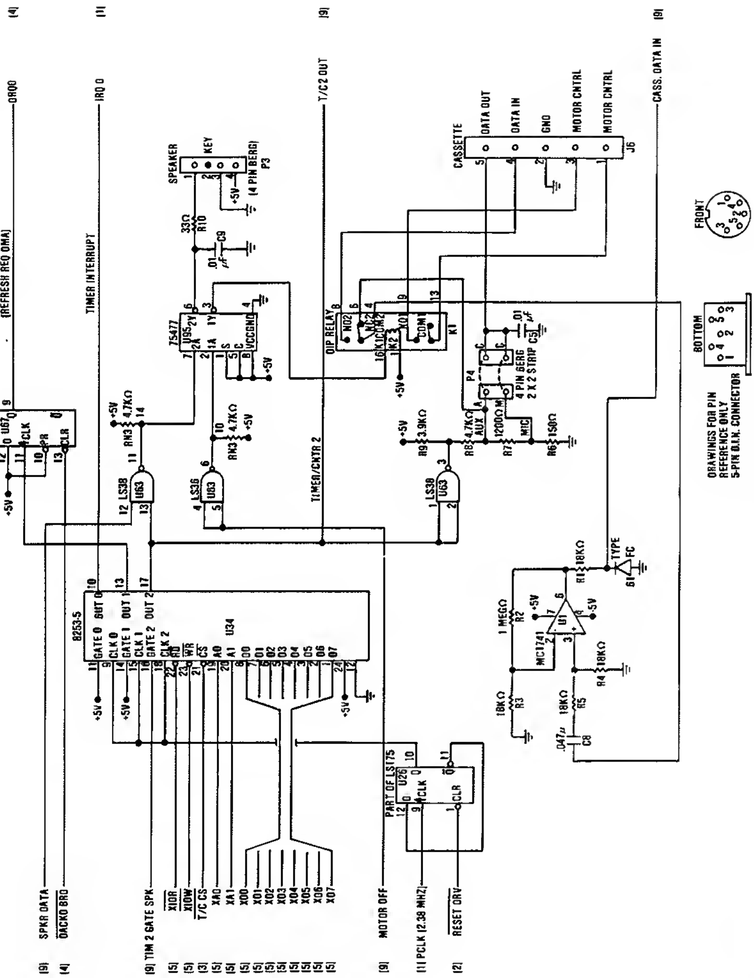
64/256K System Board (Sheet 6 of 10)

Appendix D

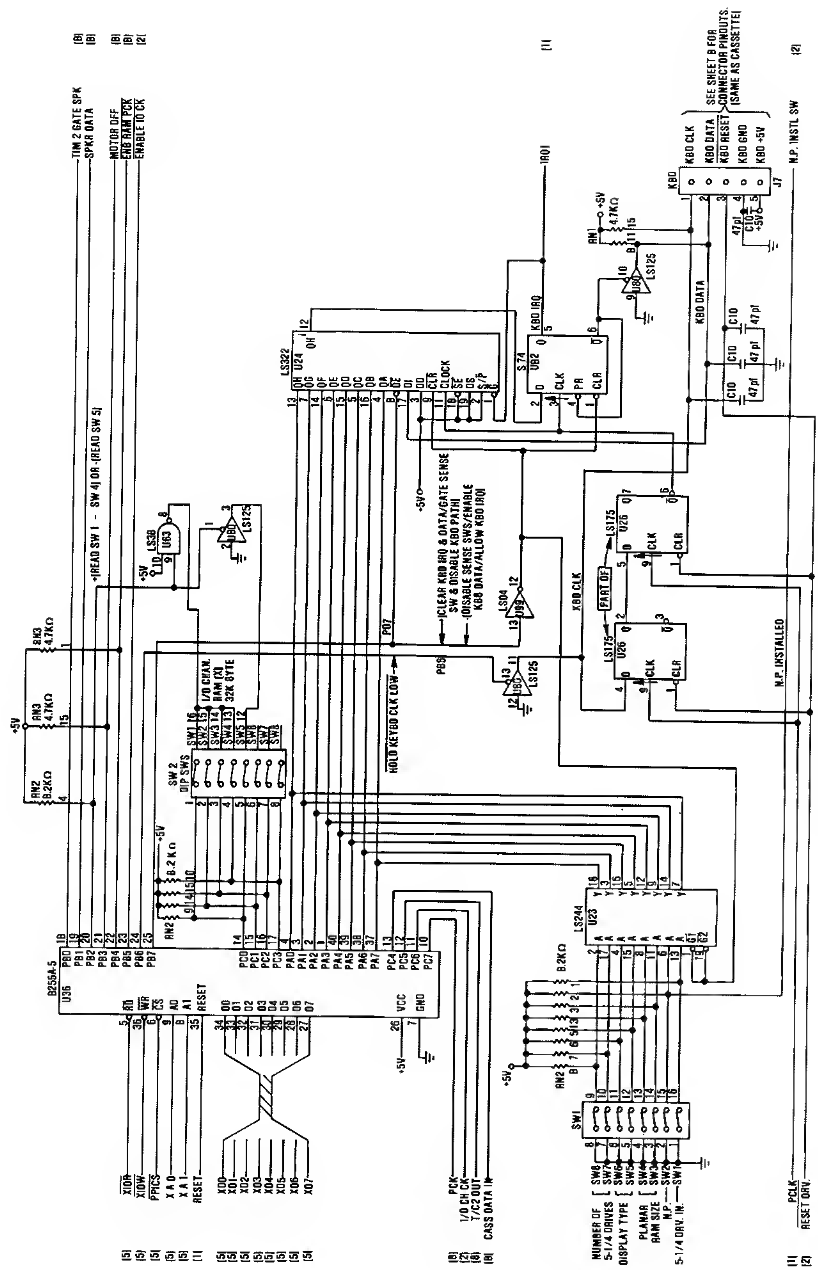


*BANKS 2 & 3 ARE FEATURES.

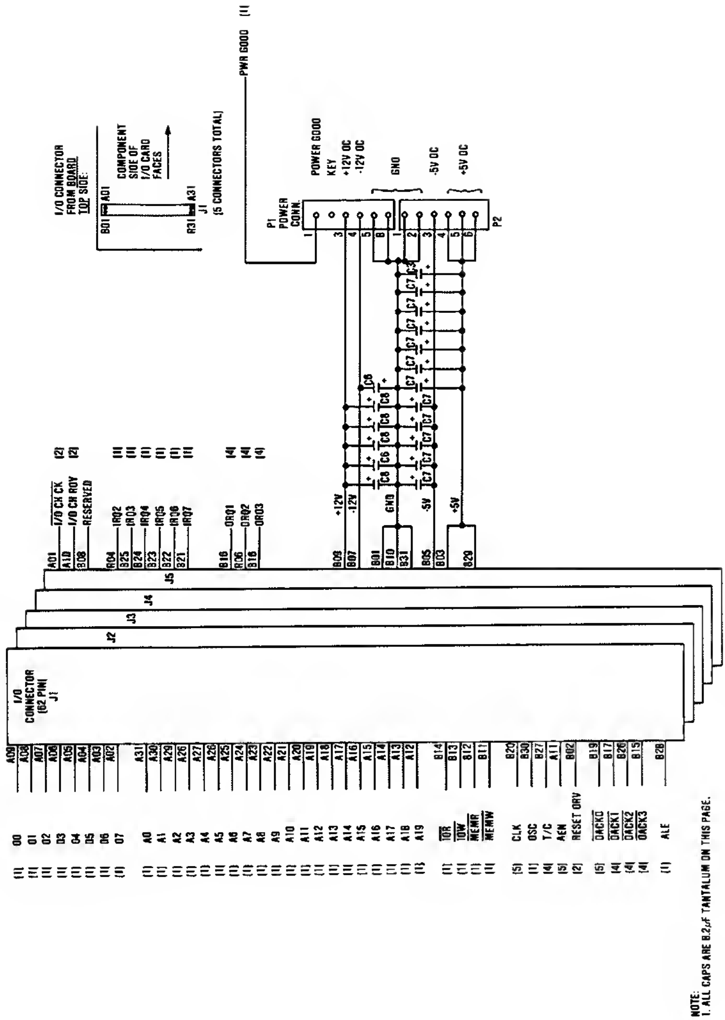
64/256K System Board (Sheet 7 of 10)



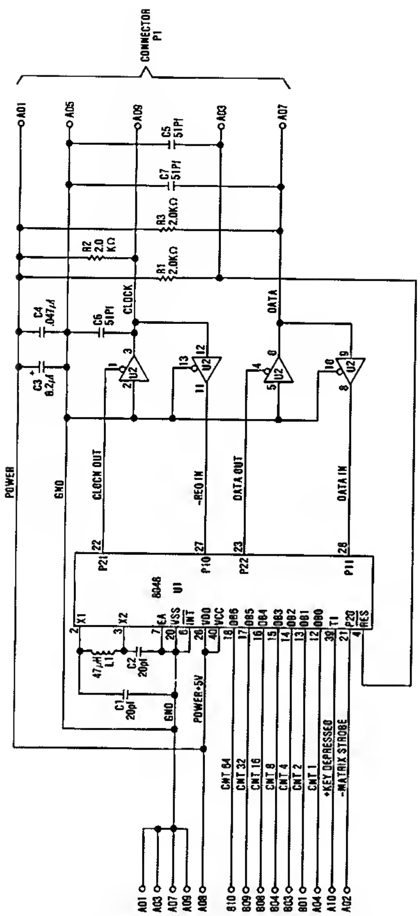
64/256K System Board (Sheet 8 of 10)



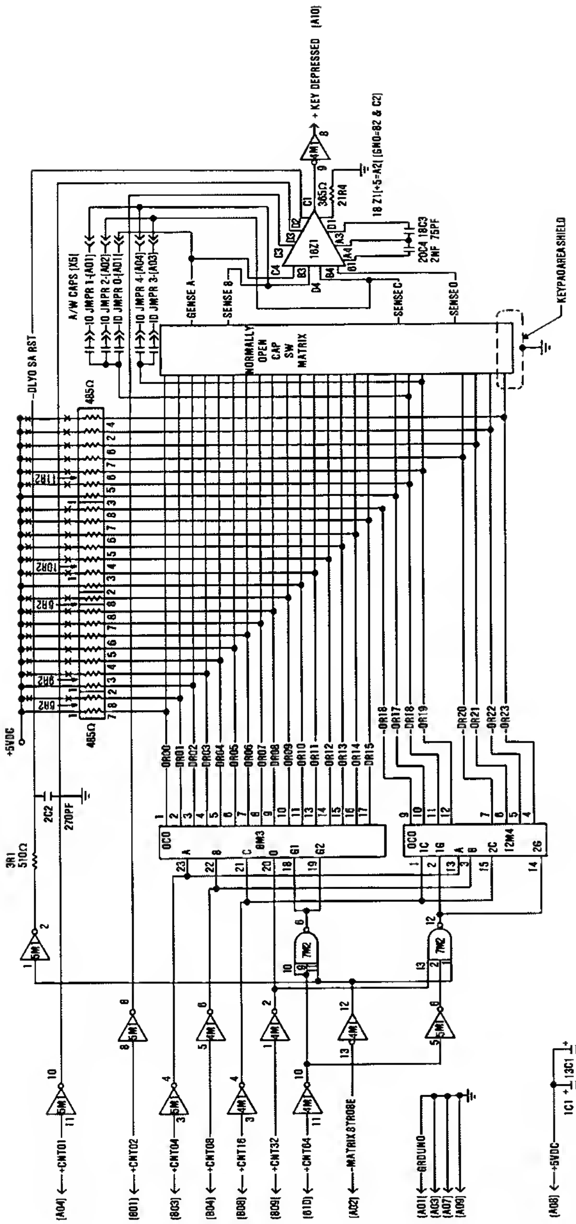
64/256K System Board (Sheet 9 of 10)



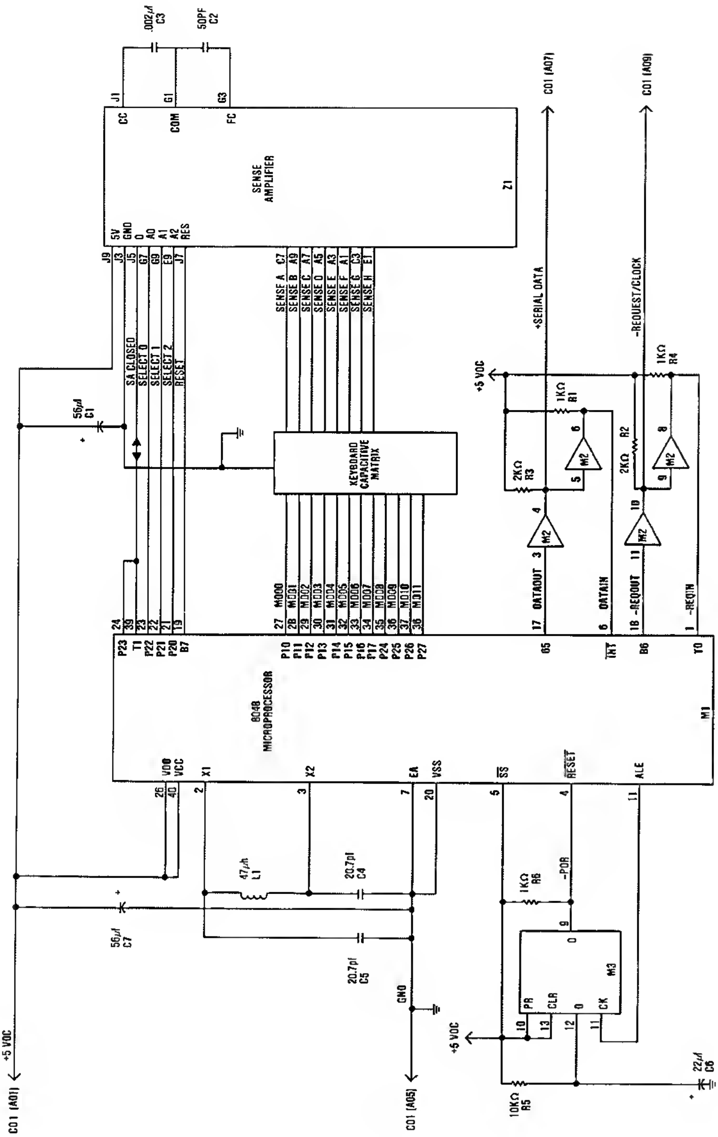
64/256K System Board (Sheet 10 of 10)



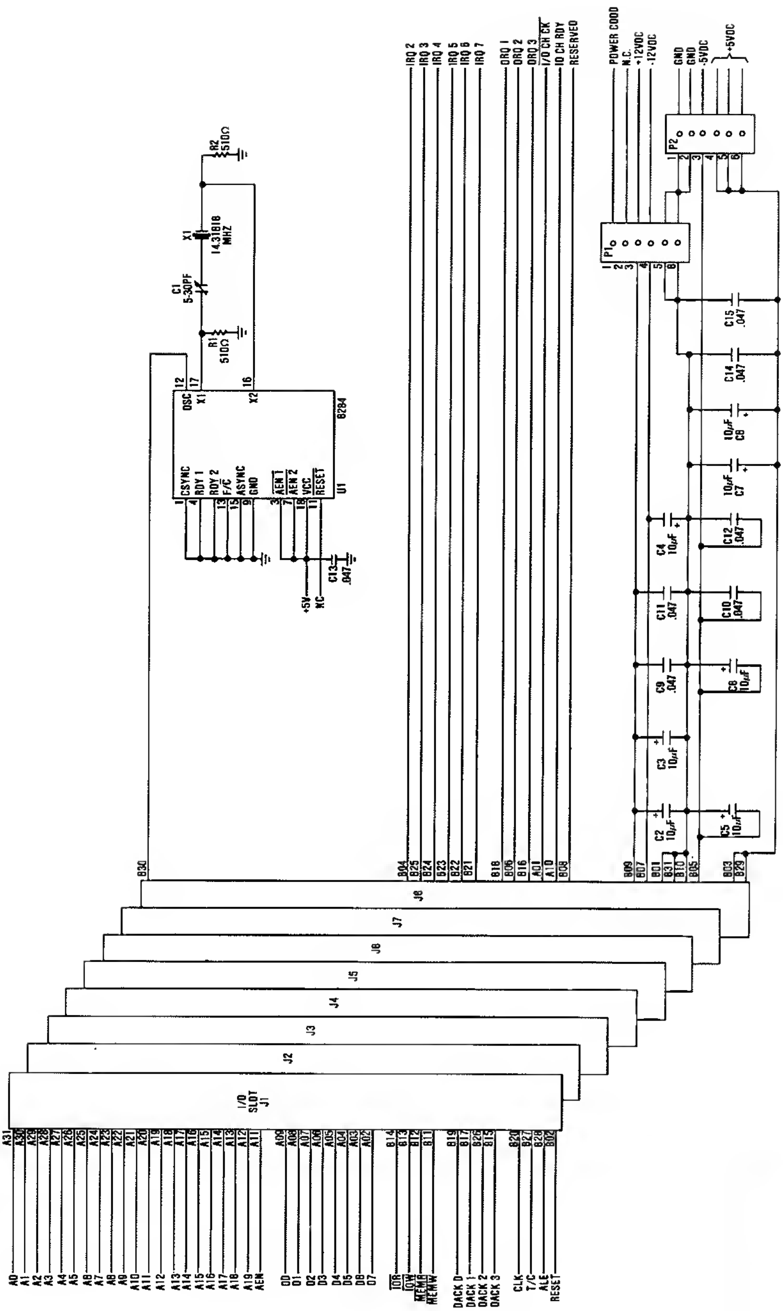
Keyboard — Type 1 (Sheet 1 of 2)



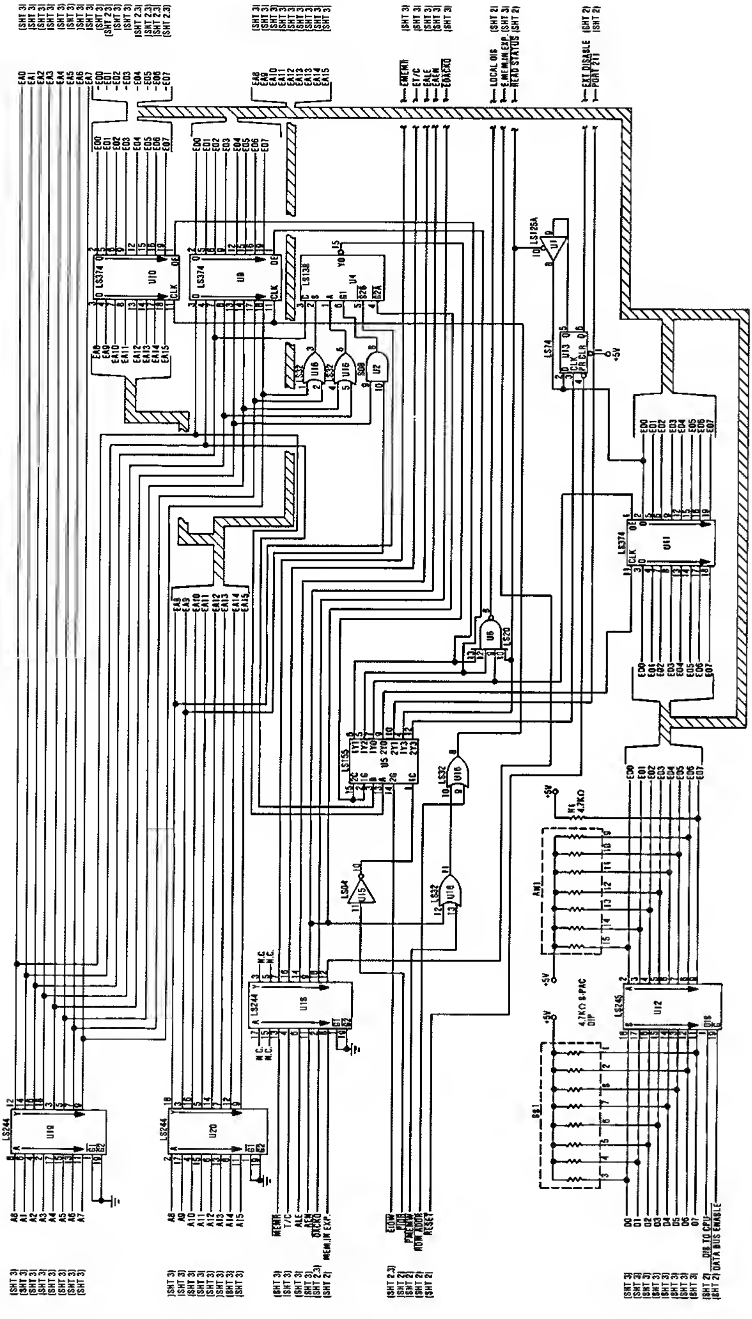
Keyboard — Type 1 (Sheet 2 of 2)



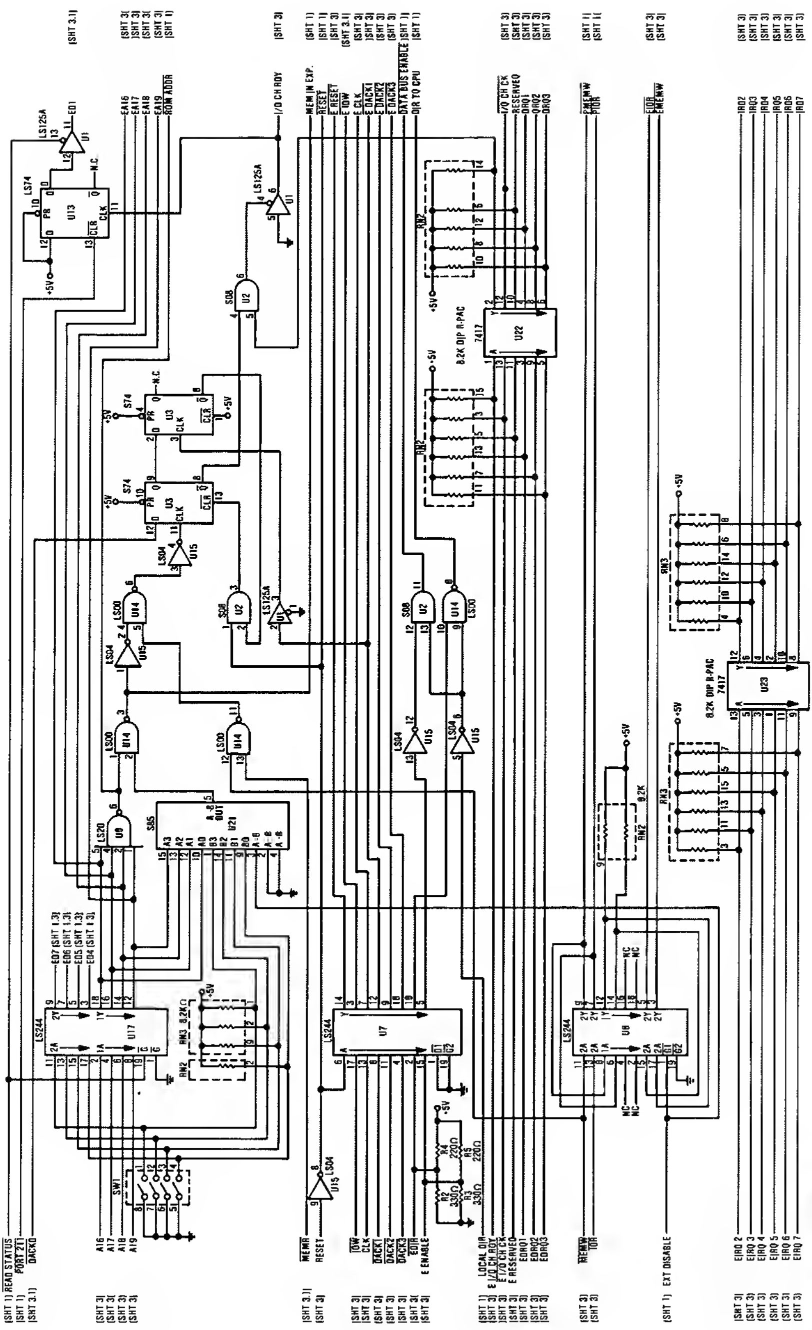
Keyboard — Type 2 (Sheet 1 of 1)



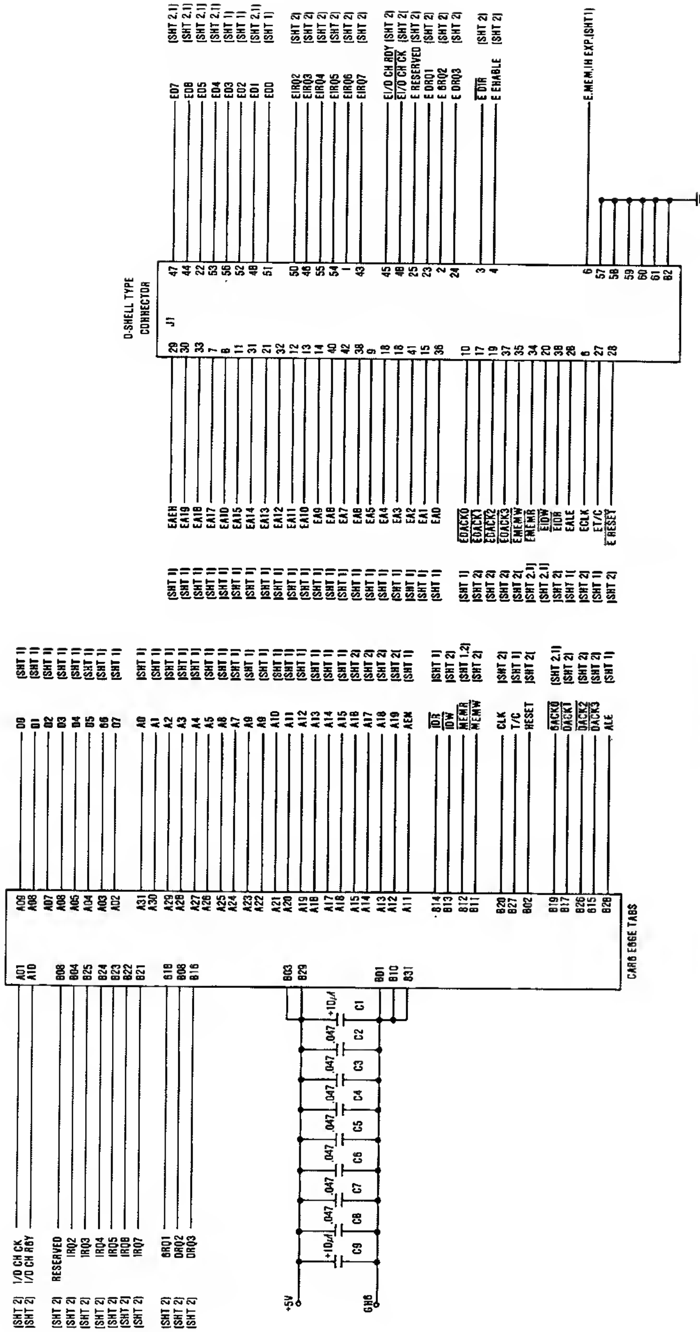
Expansion Board (Sheet 1 of 1)



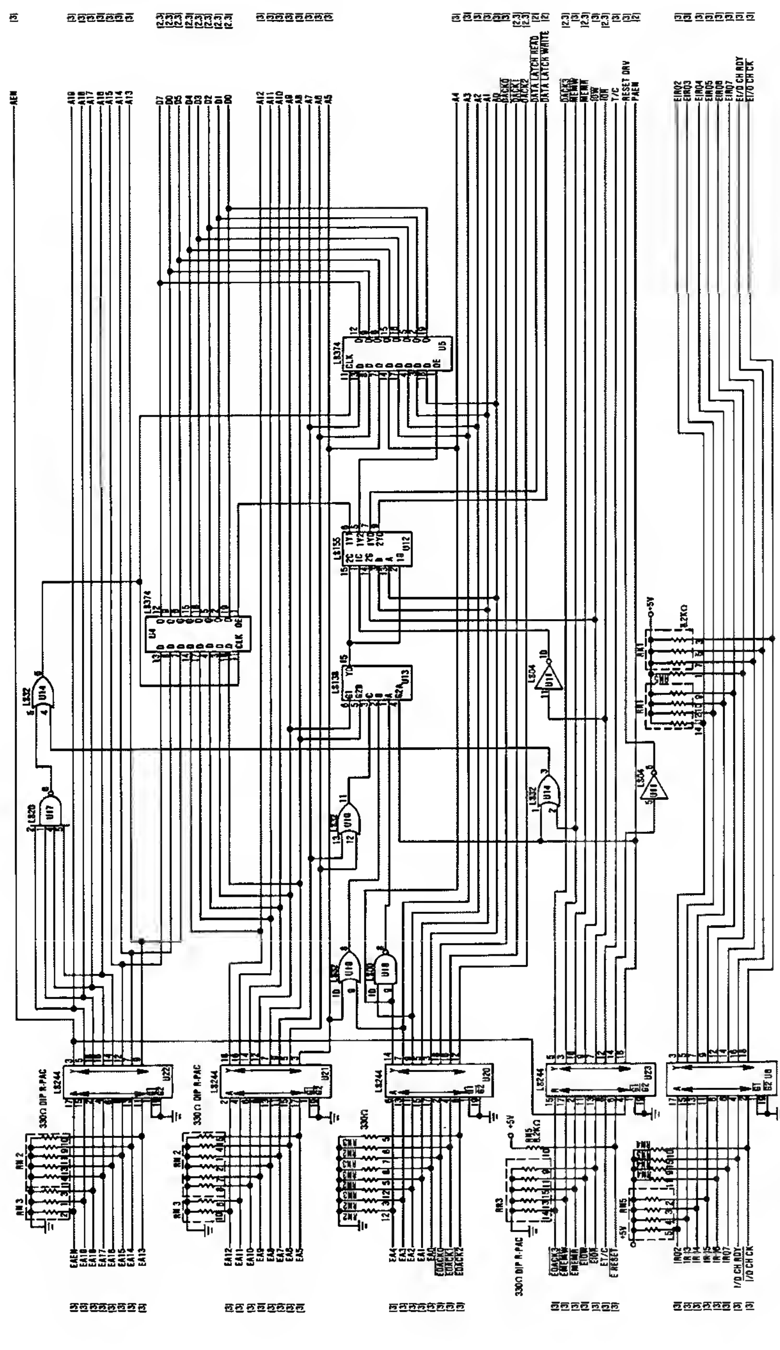
Extender Card (Sheet 1 of 3)



Extender Card (Sheet 2 of 3)

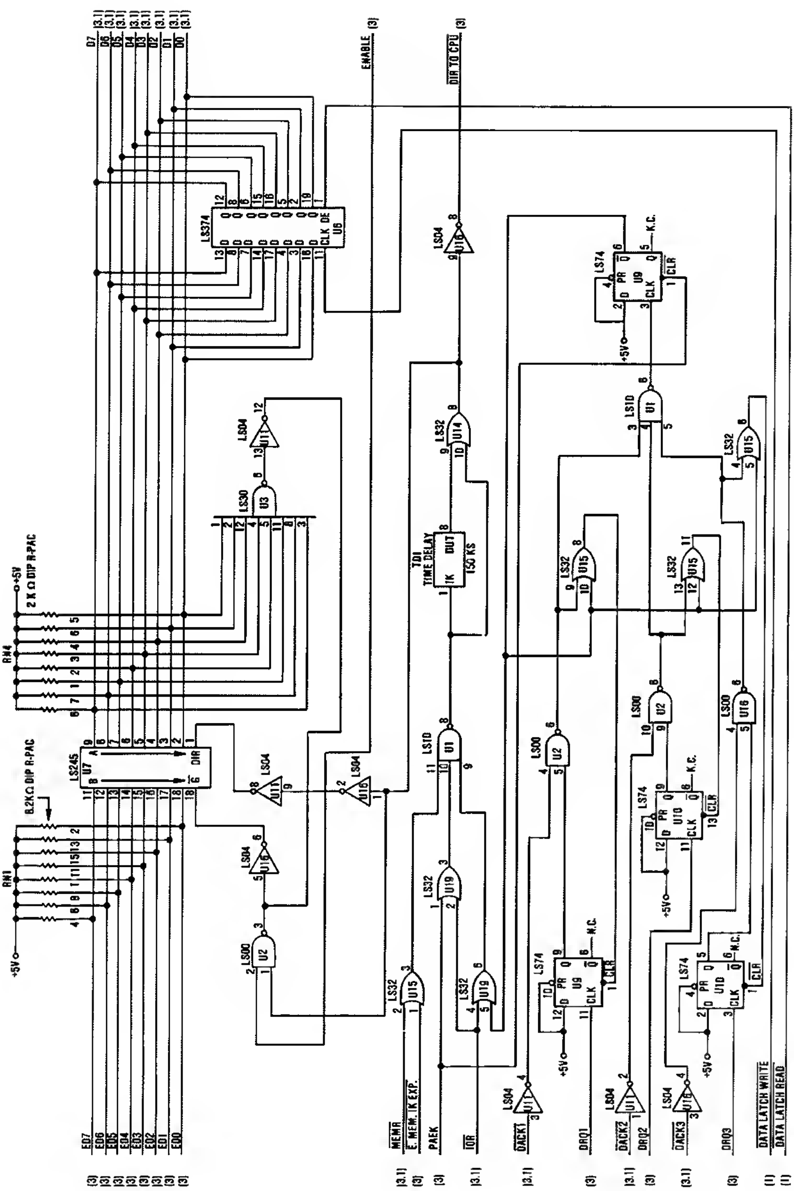


Extender Card (Sheet 3 of 3)

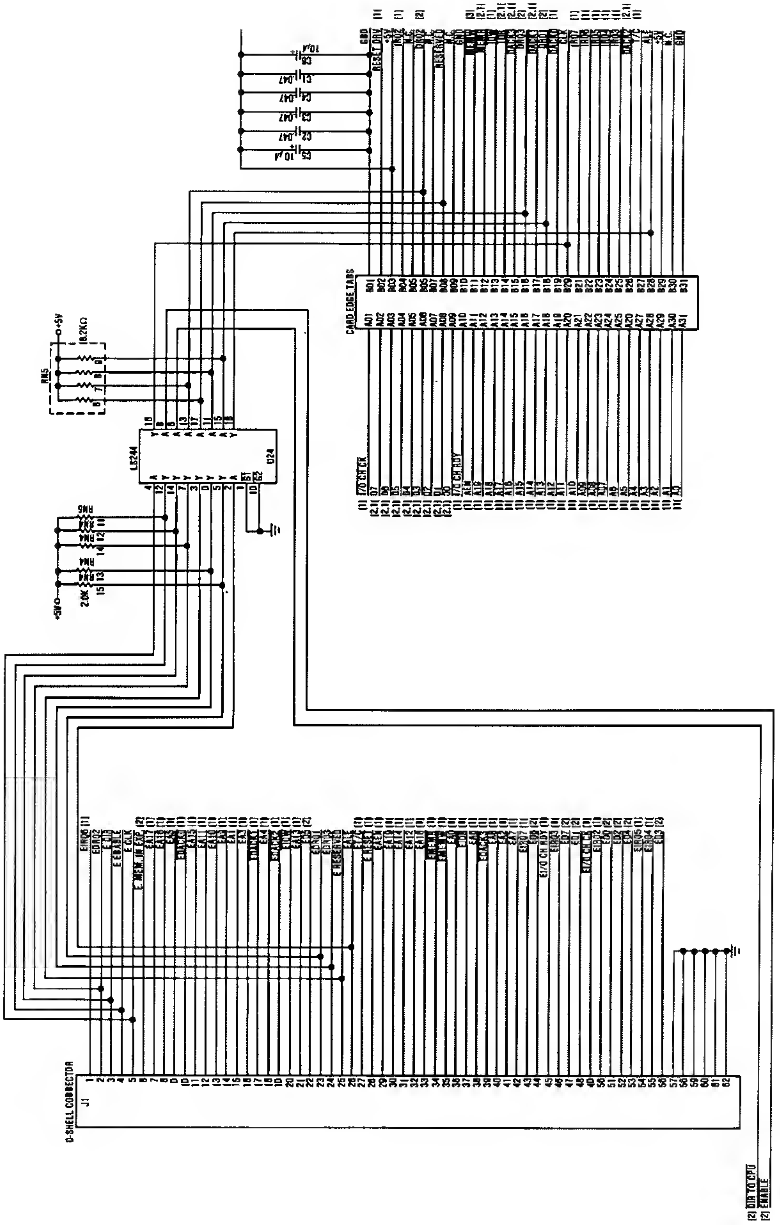


Receiver Card (Sheet 1 of 3)

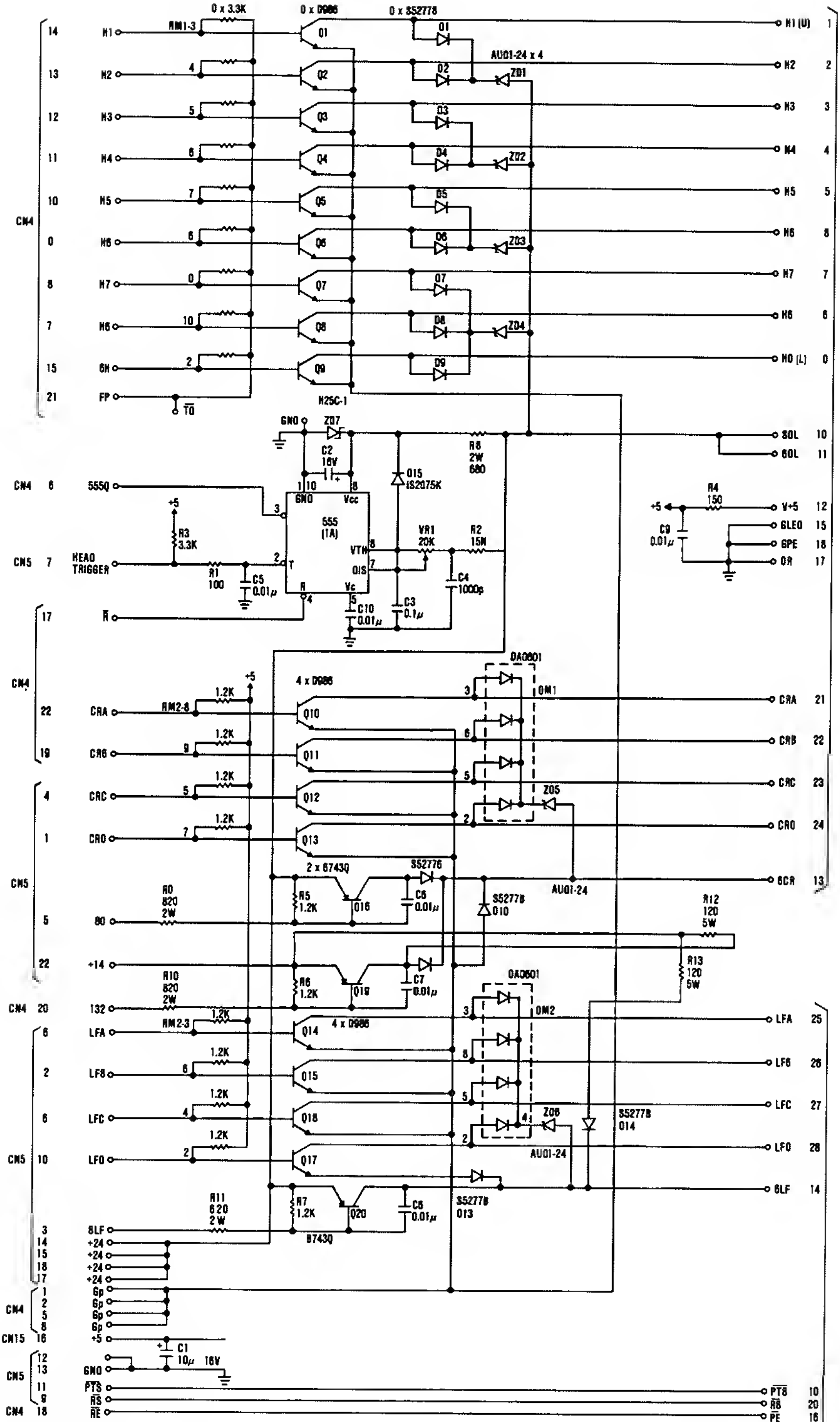
Appendix D



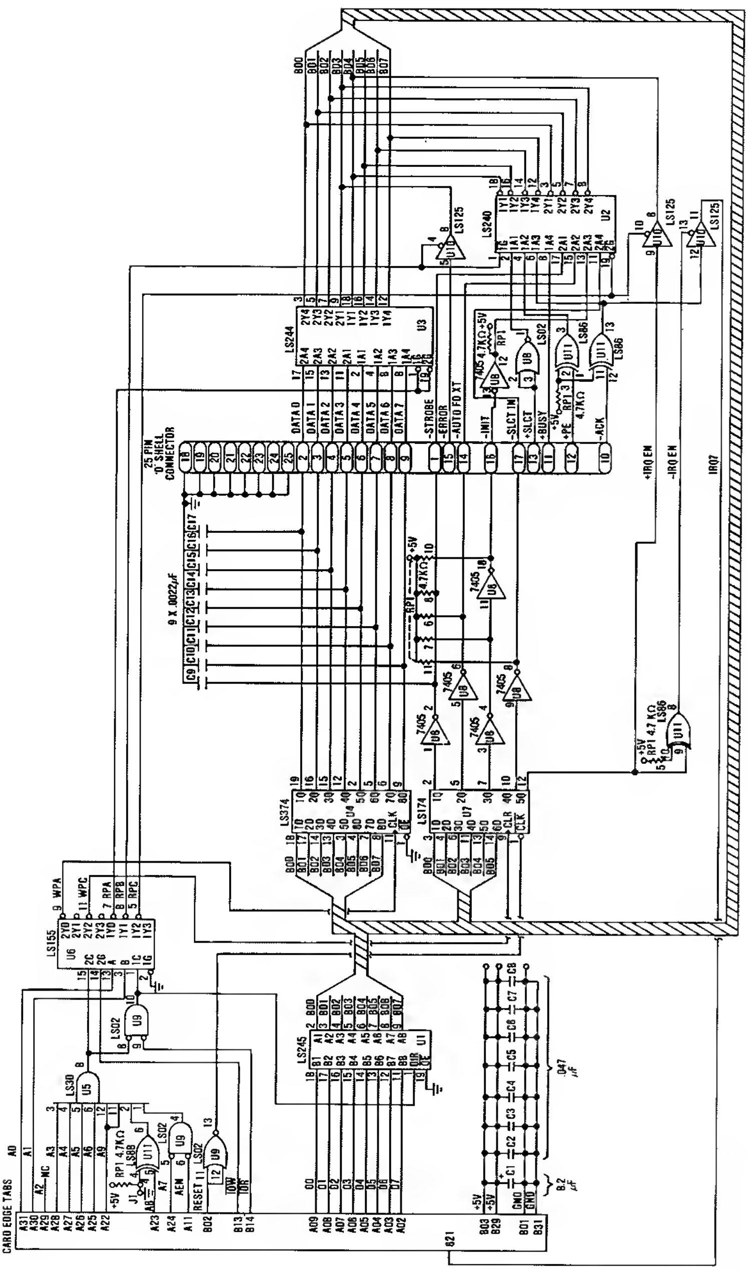
Receiver Card (Sheet 2 of 3)



Receiver Card (Sheet 3 of 3)

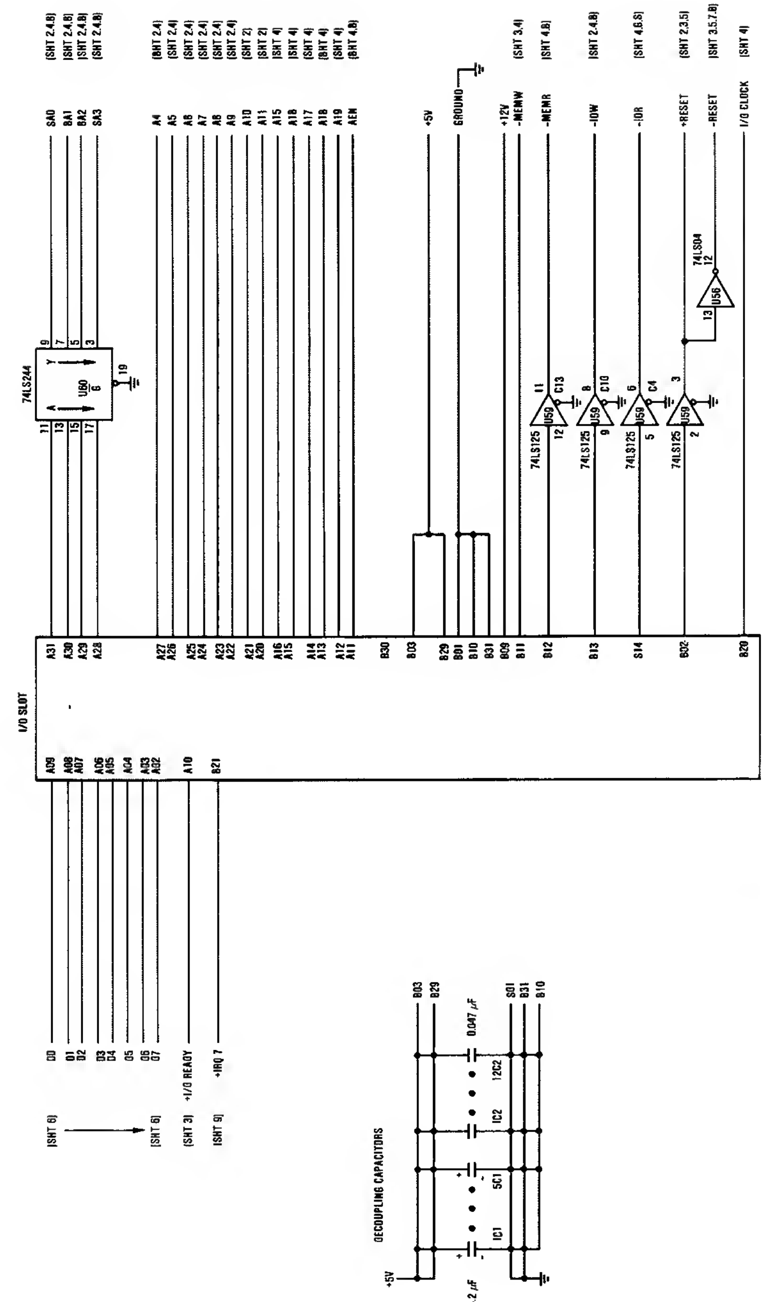


Printer (Sheet 1 of 2)

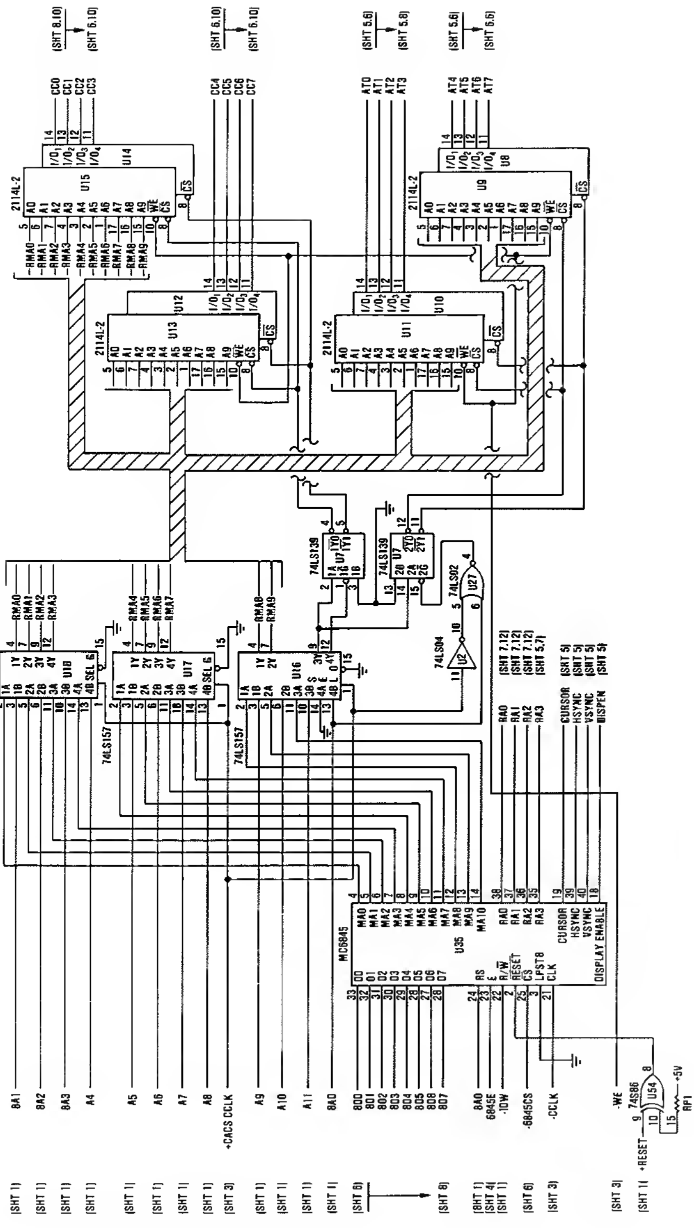


Printer Adapter (Sheet 1 of 1)

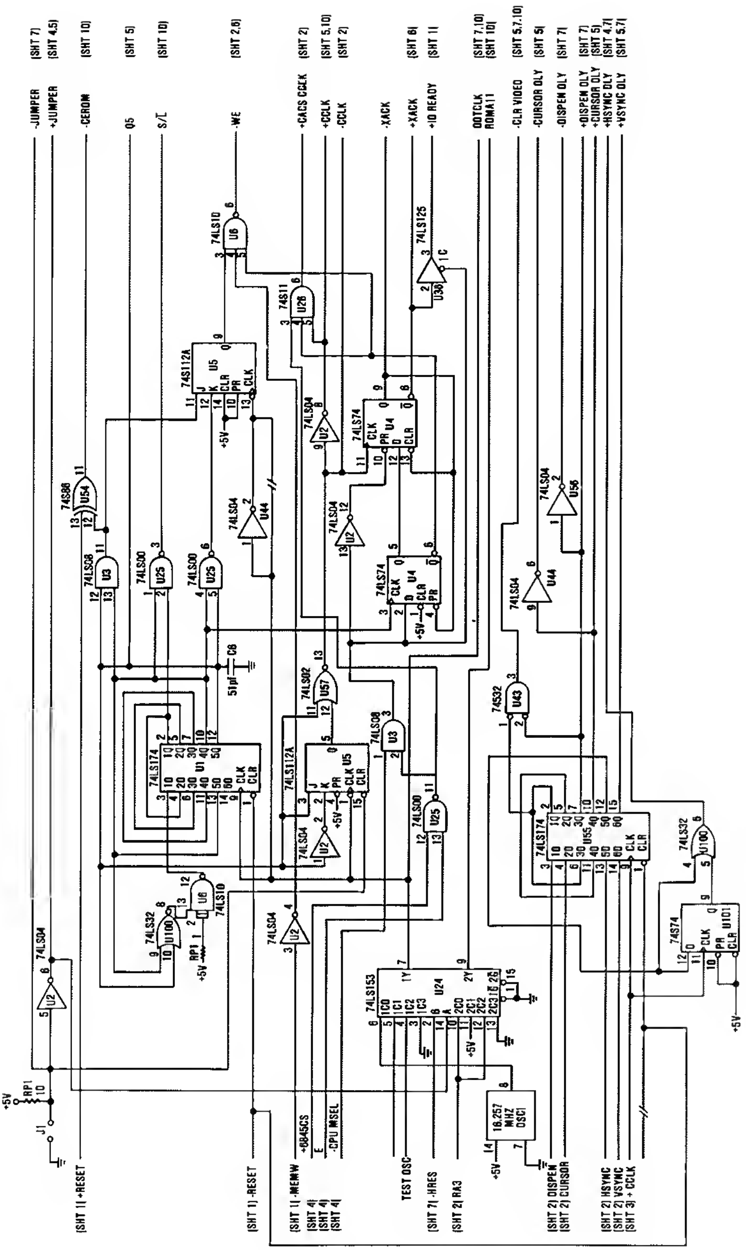
Appendix D



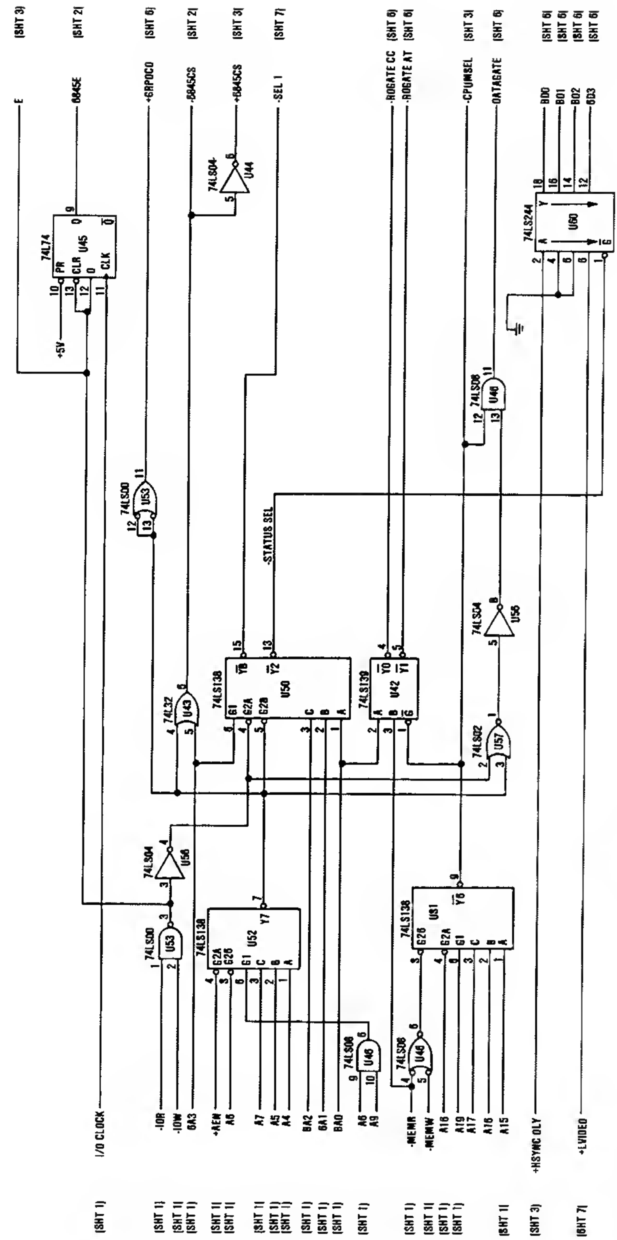
Monochrome Display Adapter (Sheet 1 of 10)



Monochrome Display Adapter (Sheet 2 of 10)

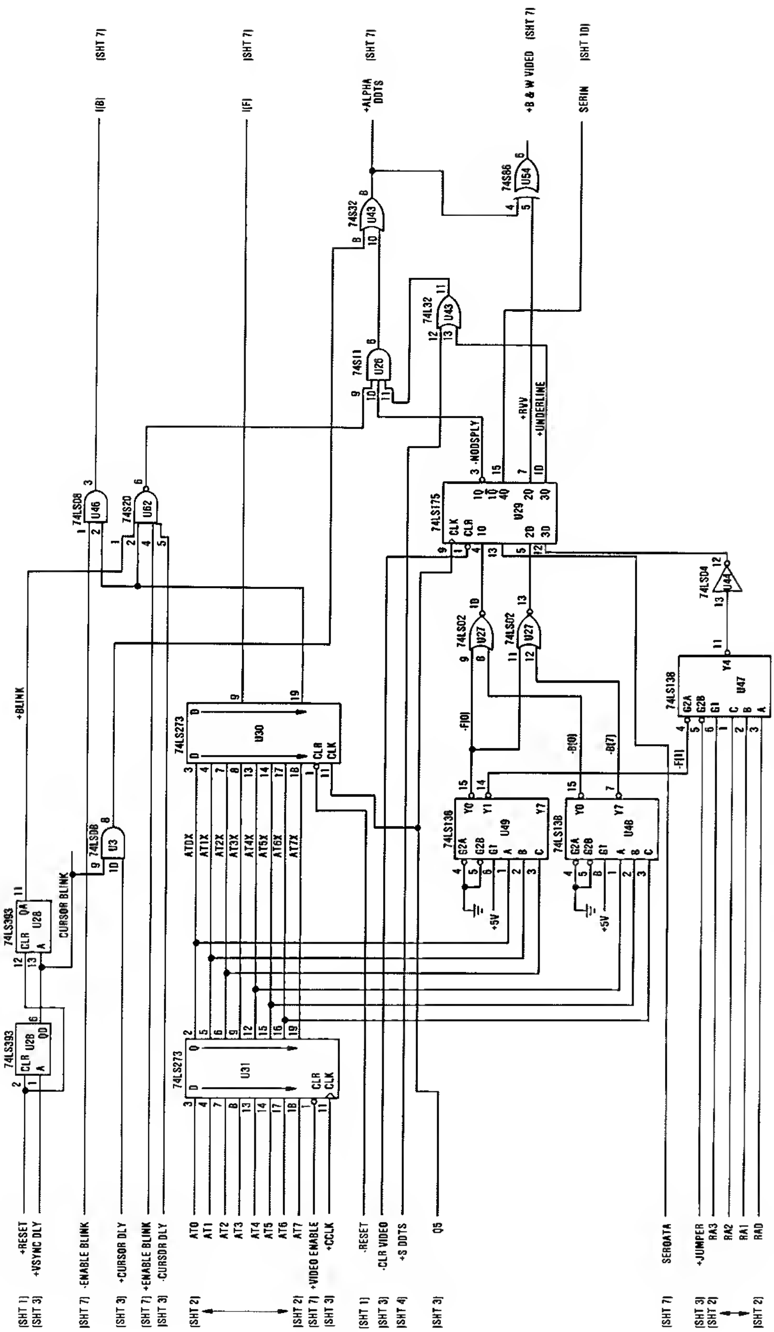


Monochrome Display Adapter (Sheet 3 of 10)

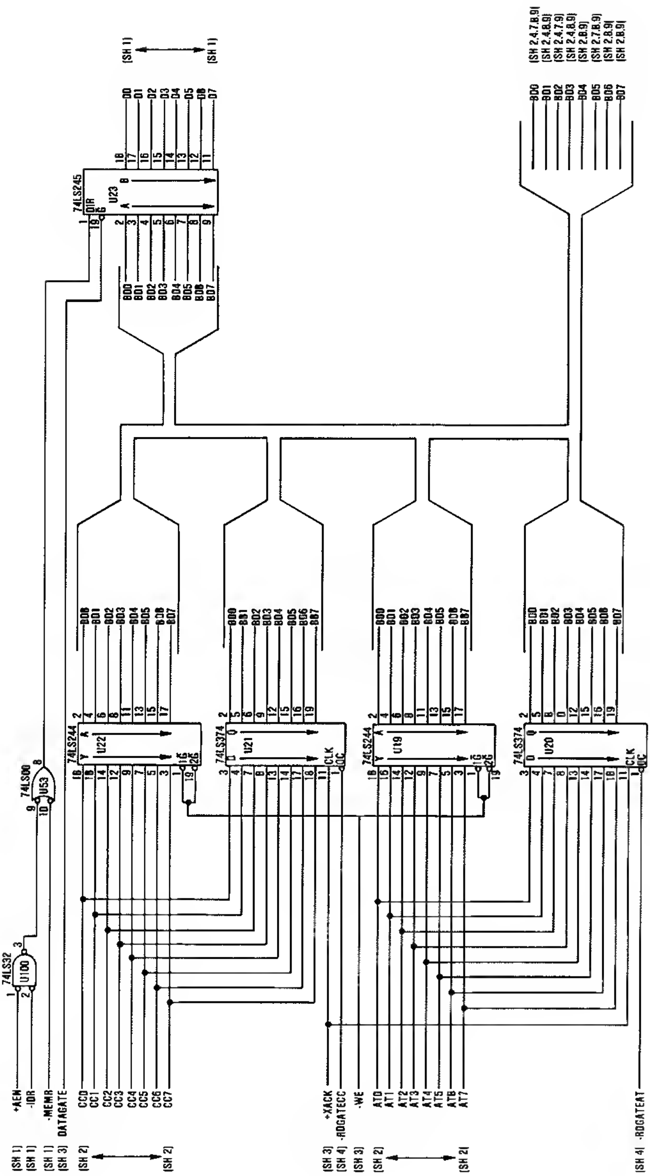


Monochrome Display Adapter (Sheet 4 of 10)

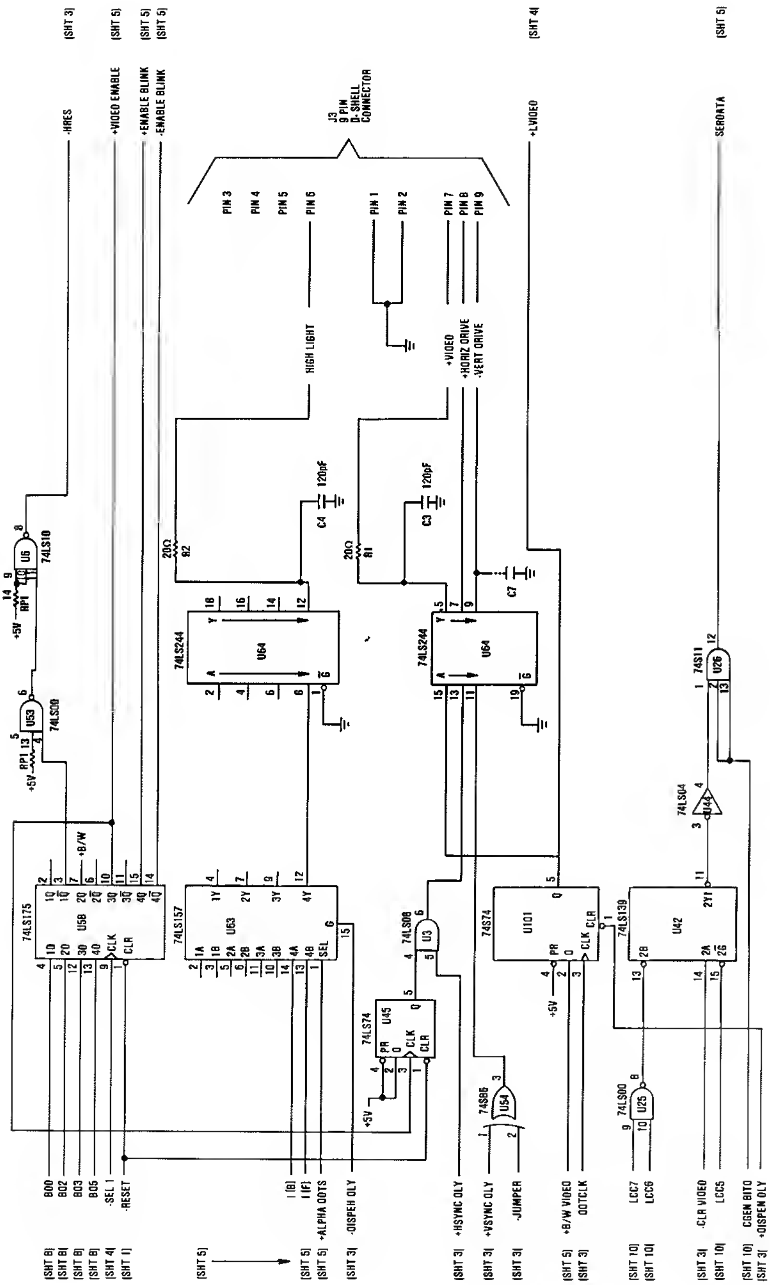
Appendix D



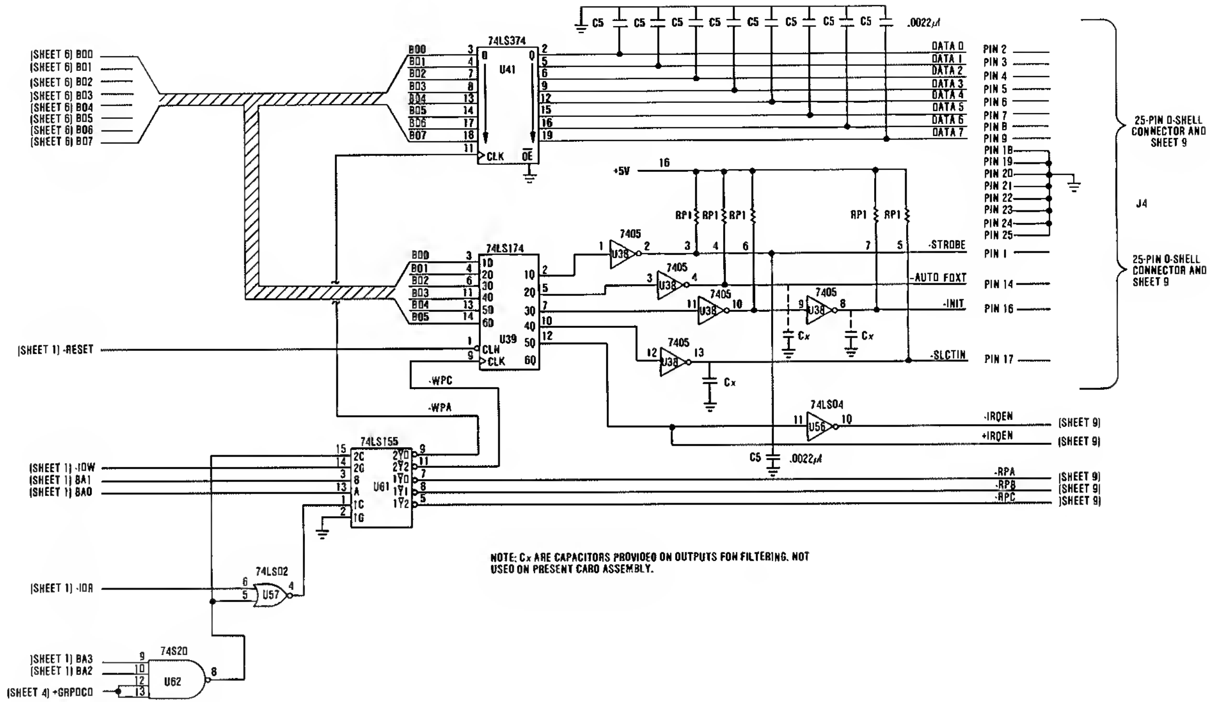
Monochrome Display Adapter (Sheet 5 of 10)



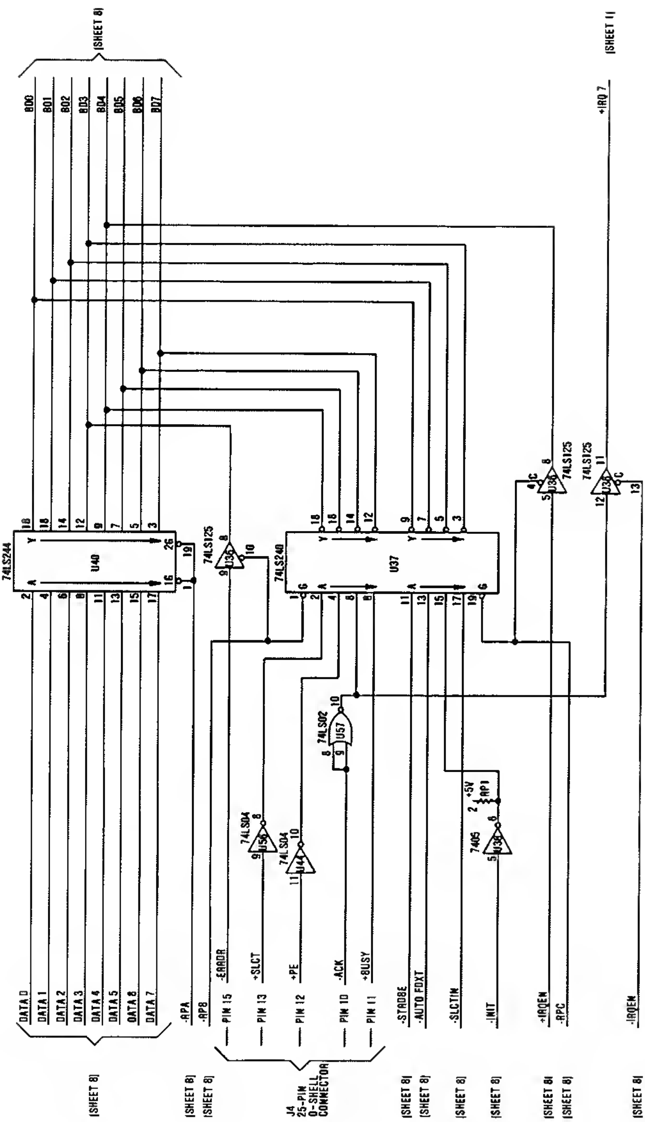
Monochrome Display Adapter (Sheet 6 of 10)



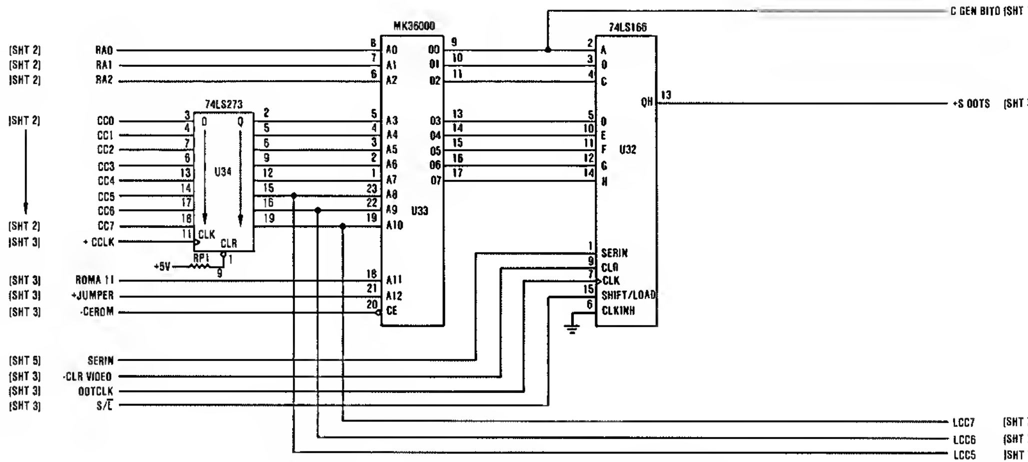
Monochrome Display Adapter (Sheet 7 of 10)



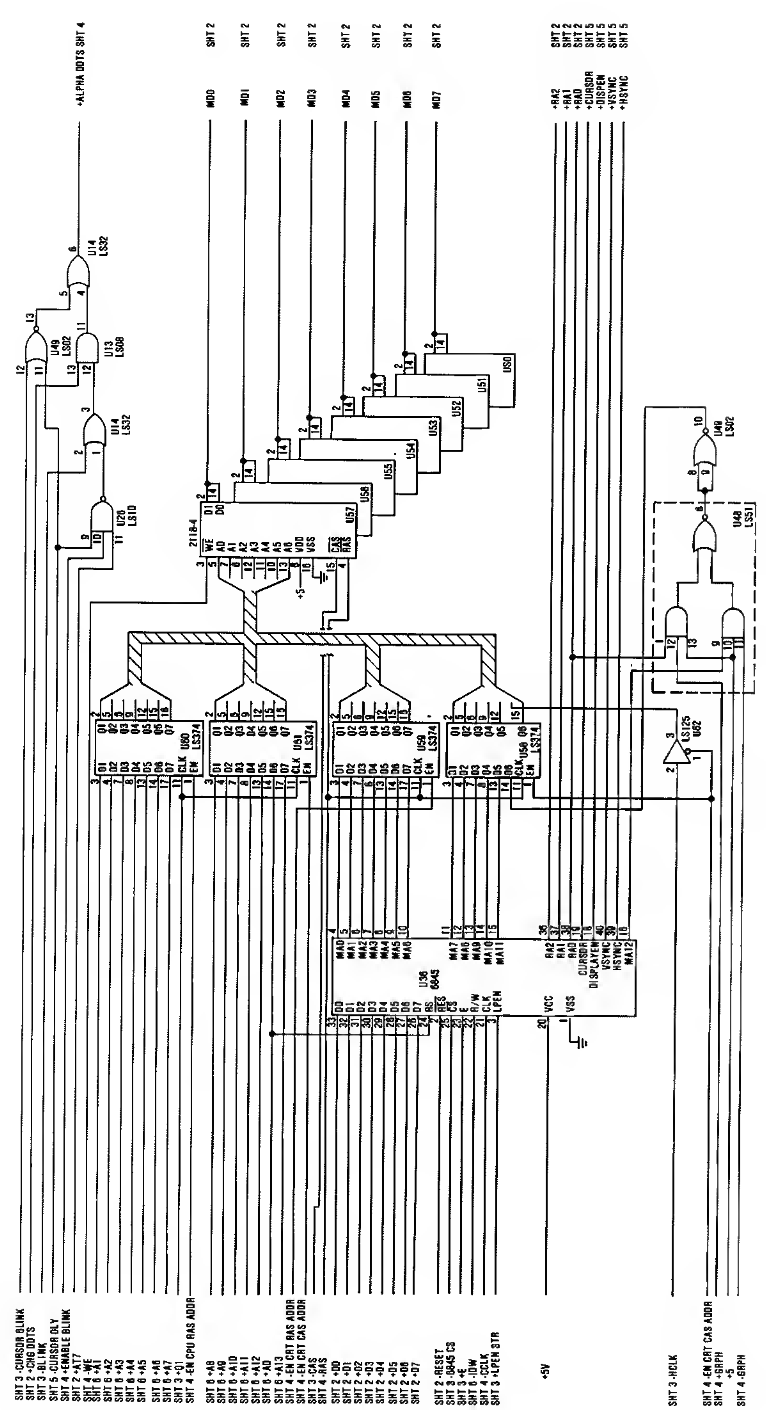
Monochrome Display Adapter (Sheet 8 of 10)



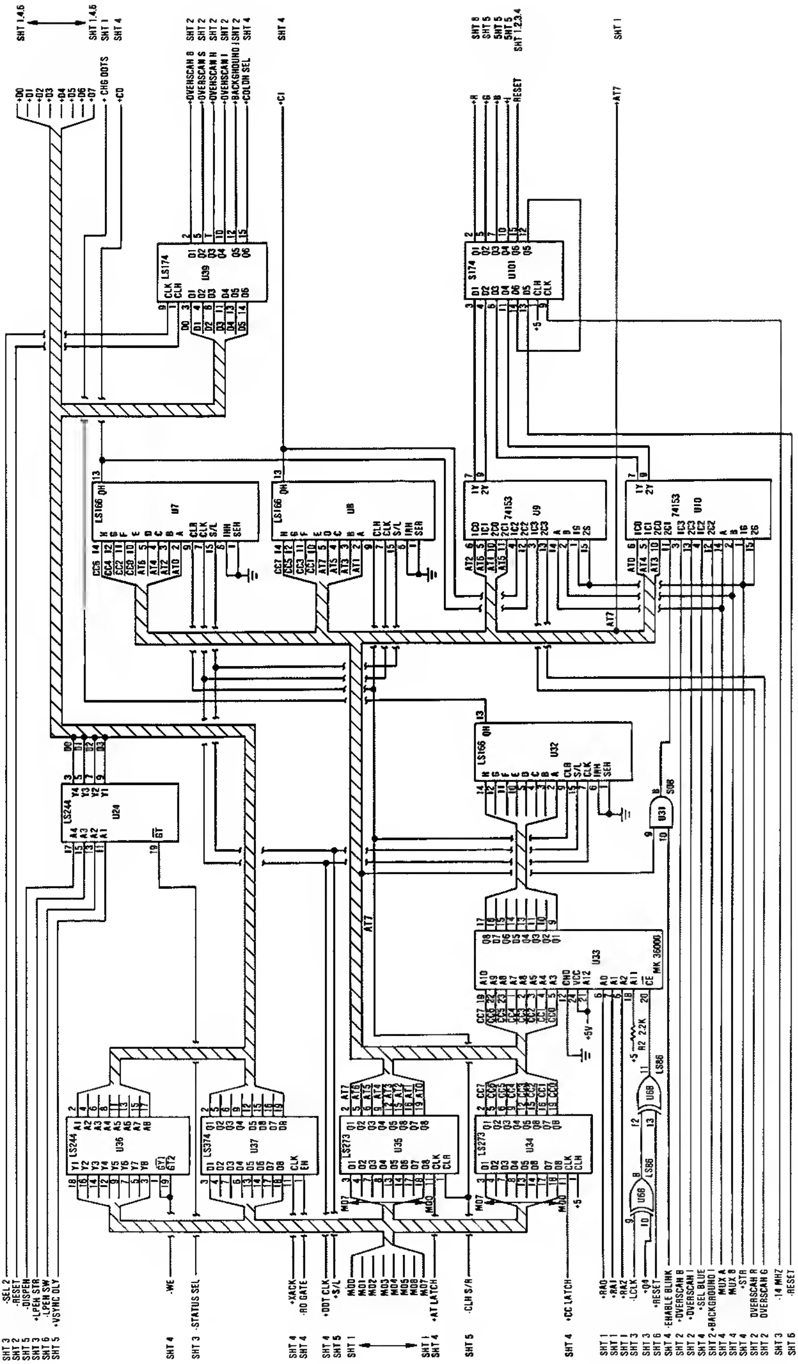
Monochrome Display Adapter (Sheet 9 of 10)



Monochrome Display Adapter (Sheet 10 of 10)

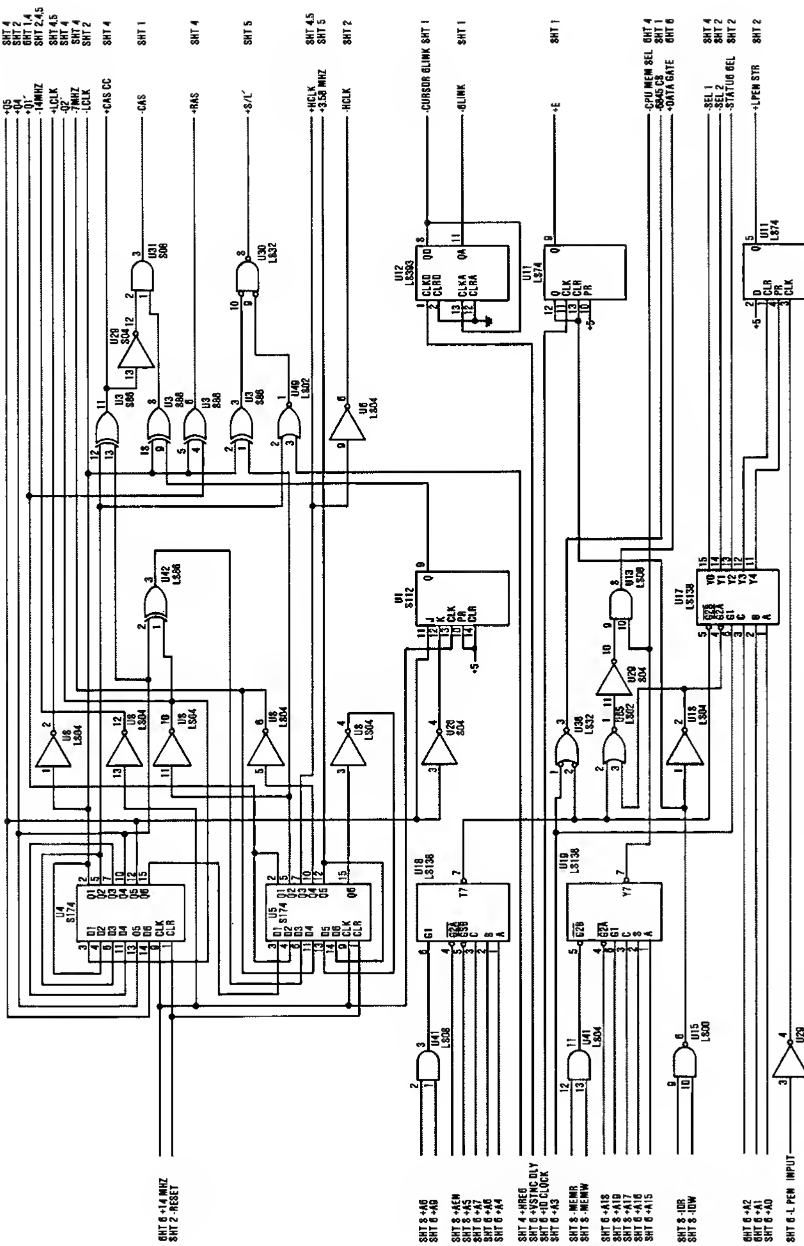


Color/Graphics Monitor Adapter (Sheet 1 of 6)

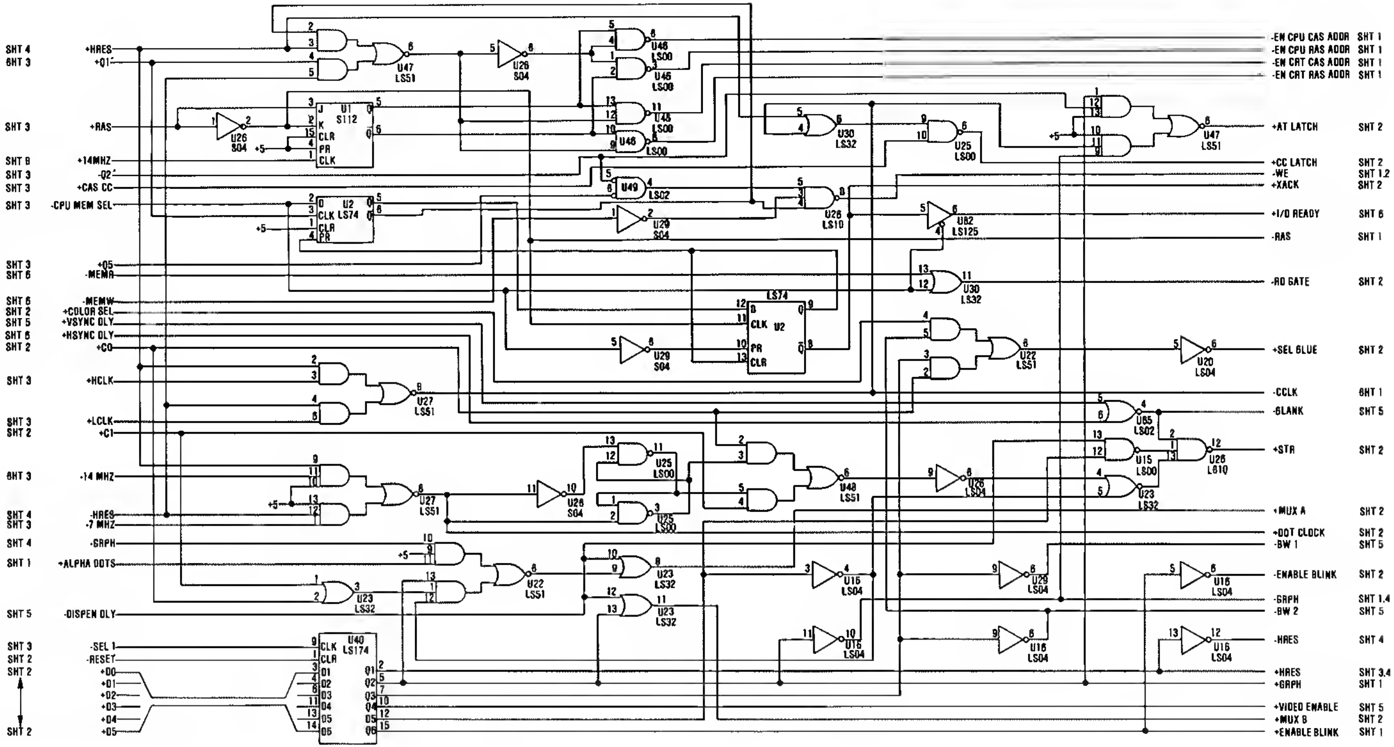


Color/Graphics Monitor Adapter (Sheet 2 of 6)

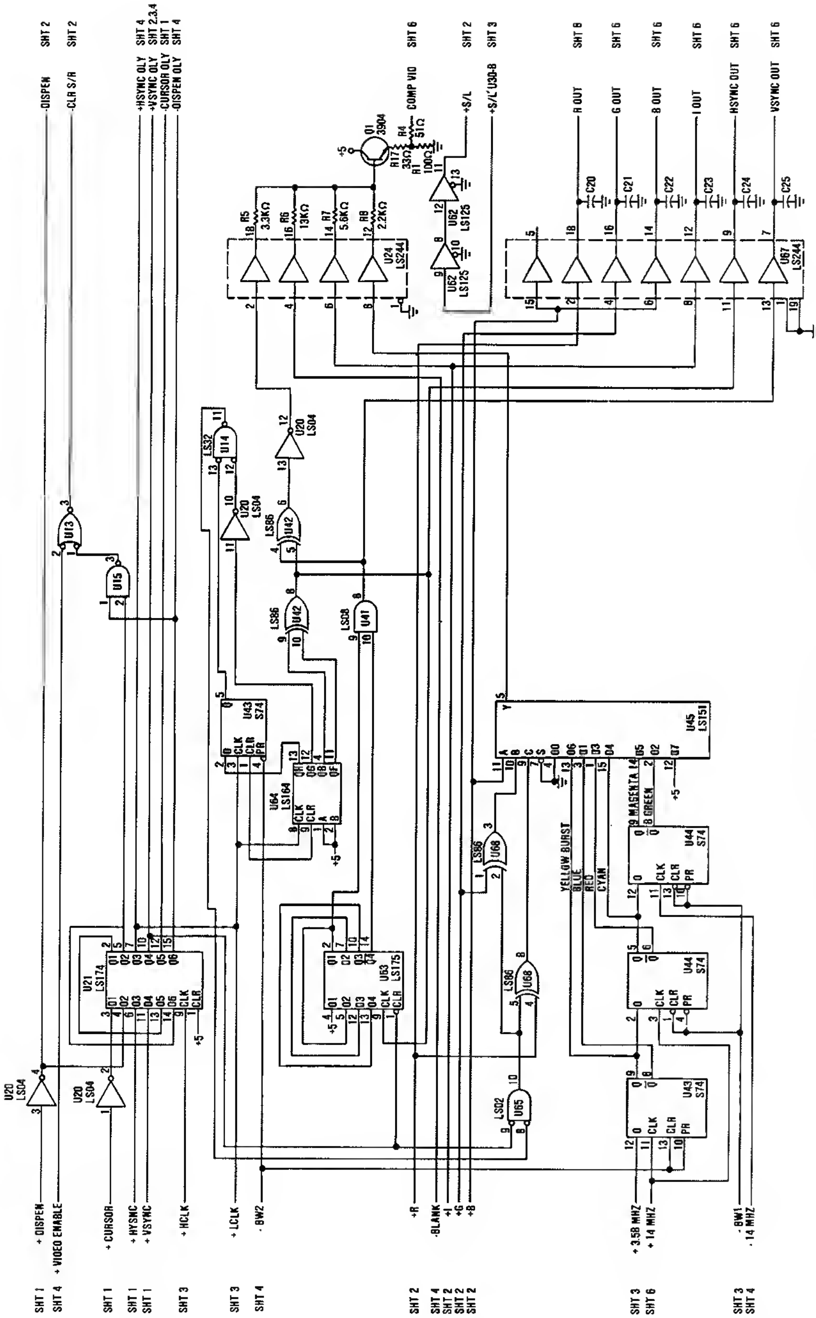
Appendix D



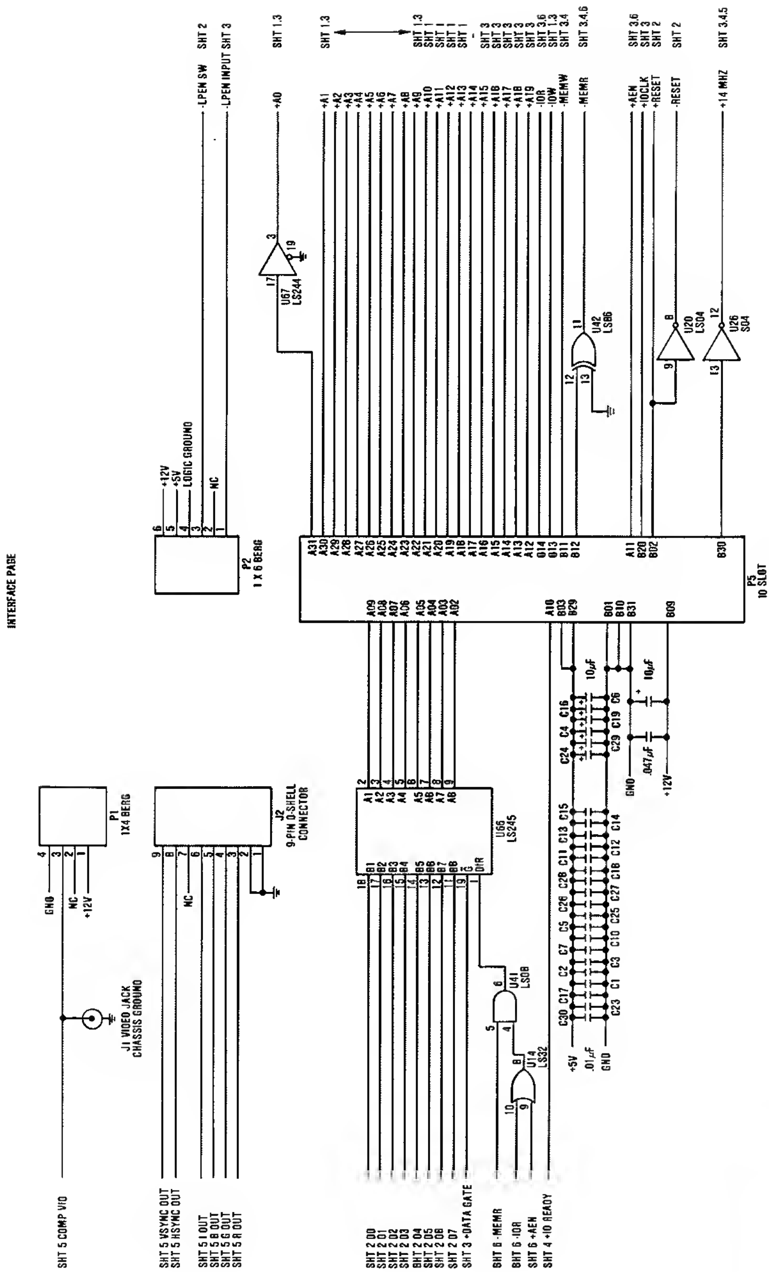
Color/Graphics Monitor Adapter (Sheet 3 of 6)



Color/Graphics Monitor Adapter (Sheet 4 of 6)



Color/Graphics Monitor Adapter (Sheet 5 of 6)

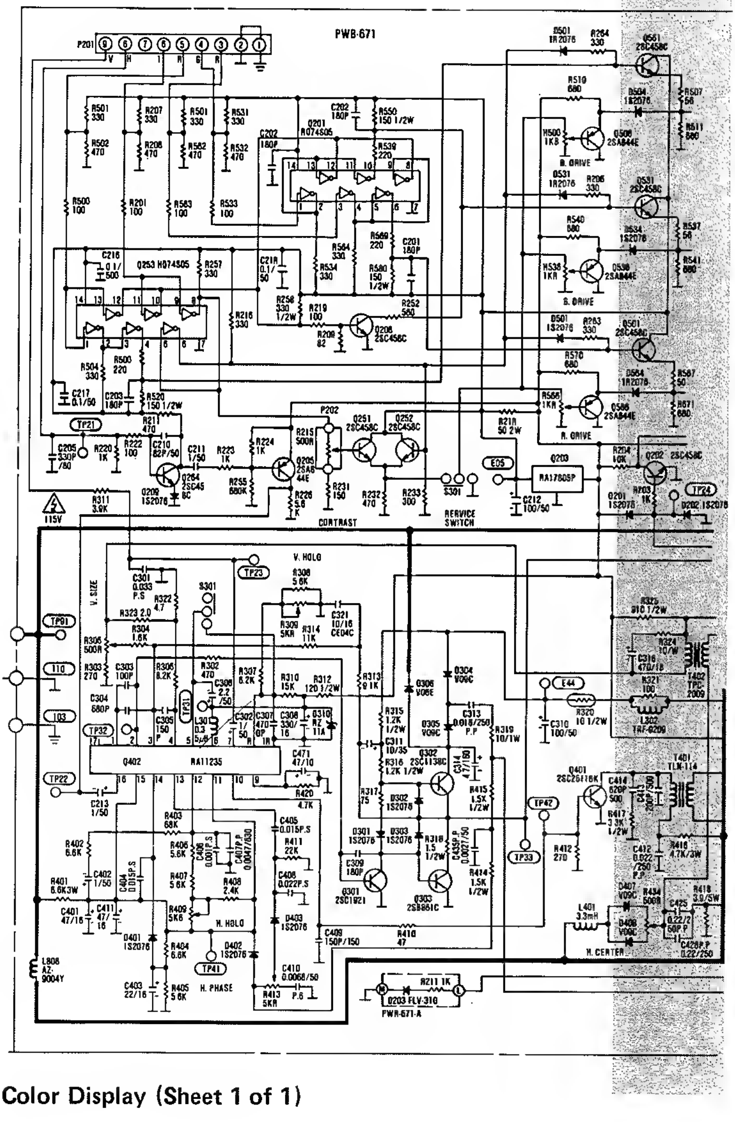


Color/Graphics Monitor Adapter (Sheet 6 of 6)

Appendix D

DANGER

HAZARDOUS VOLTAGES UP TO 450 VOLTS EXIST ON THE PRINTED CIRCUIT BOARDS

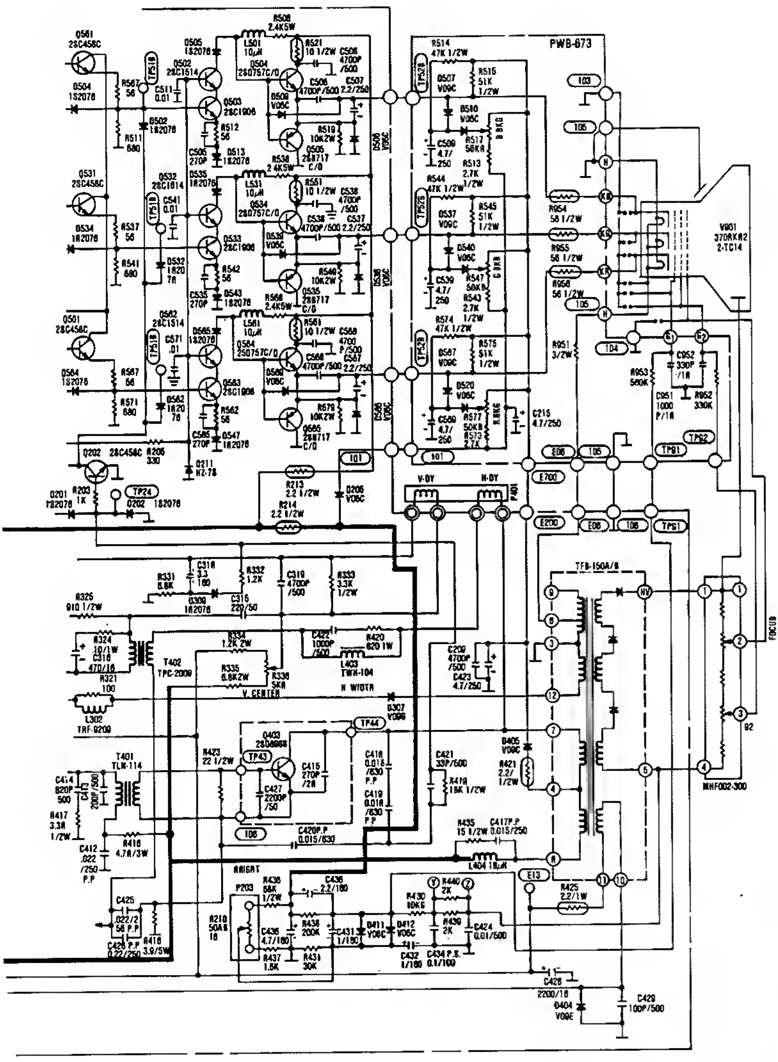


Color Display (Sheet 1 of 1)

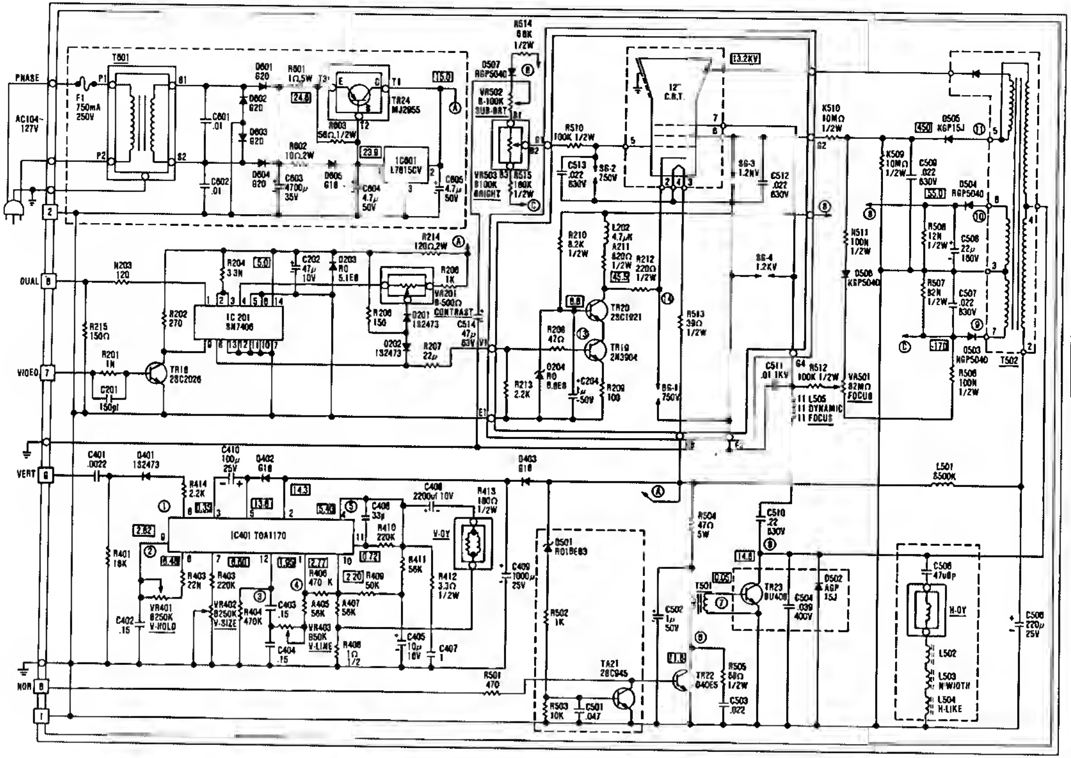
DANGER

HAZARDOUS VOLTAGES UP TO 450 VOLTS EXIST ON THE PRINTED CIRCUIT BOARDS

NOTES:
1. RESISTOR VALUES ARE IN OHMS K = 1000 OHMS.
2. ALL RESISTORS ARE 1/2 WATT EXCEPT WHERE OTHERWISE INDICATED.
3. CAPACITOR VALUES ARE IN μF UNLESS OTHERWISE INDICATED P = PF.
4. ALL CAPACITORS ARE 50 VOLTS UNLESS OTHERWISE INDICATED.



Color Display (Sheet 1 of 1)



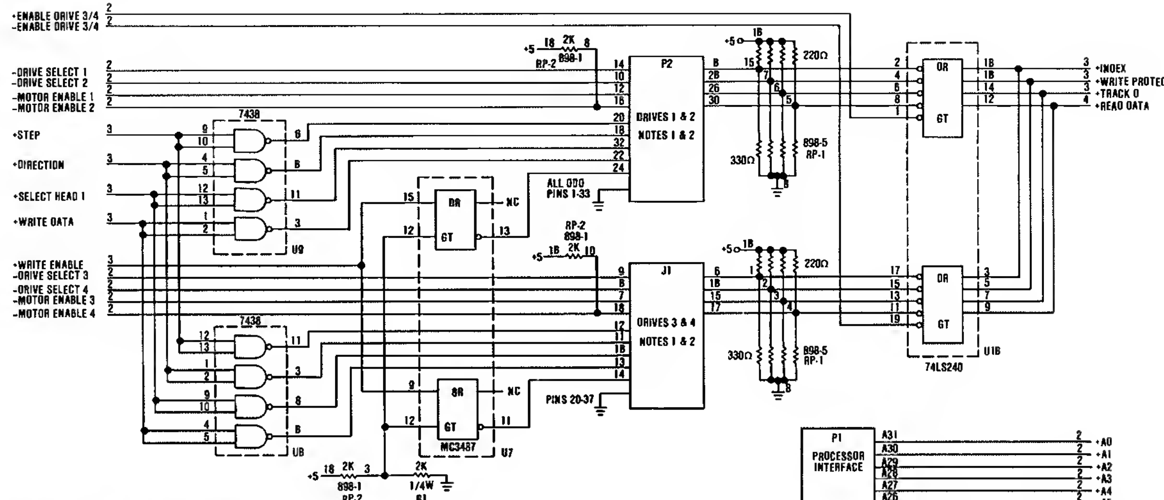
DANGER

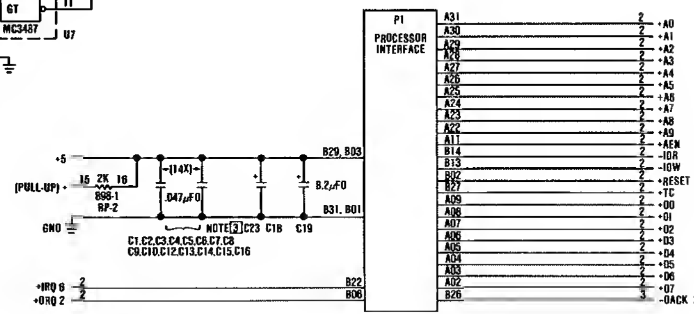
HAZARDOUS VOLTAGES UP TO 450 VOLTS EXIST ON THE PRINTED CIRCUIT BOARDS

NOTES:
1. RESISTOR VALUES ARE IN (OHM) Ω K = 1000Ω M = 1,000,000Ω
2. ALL RESISTOR ARE 1/4W EXCEPT WHERE OTHERWISE INDICATED.
3. ALL CAPACITORS ARE 50V EXCEPT WHERE OTHERWISE INDICATED
4. CAPACITORS VALUES ARE μF UNLESS OTHERWISE INDICATED μ = μF = 10⁻⁶
5. AC WIRING INFORMATION
   PHASE = BLACK/BROWN WIRE
   NEUTRAL = WHITE/BLUE WIRE
   GROUND = GREEN AND YELLOW WIRE

IMPORTANT: THE PHASE WIRE MUST GO TO THE FUSED SIDE OF TRANSFORMER

Monochrome Display (Sheet 1 of 1)





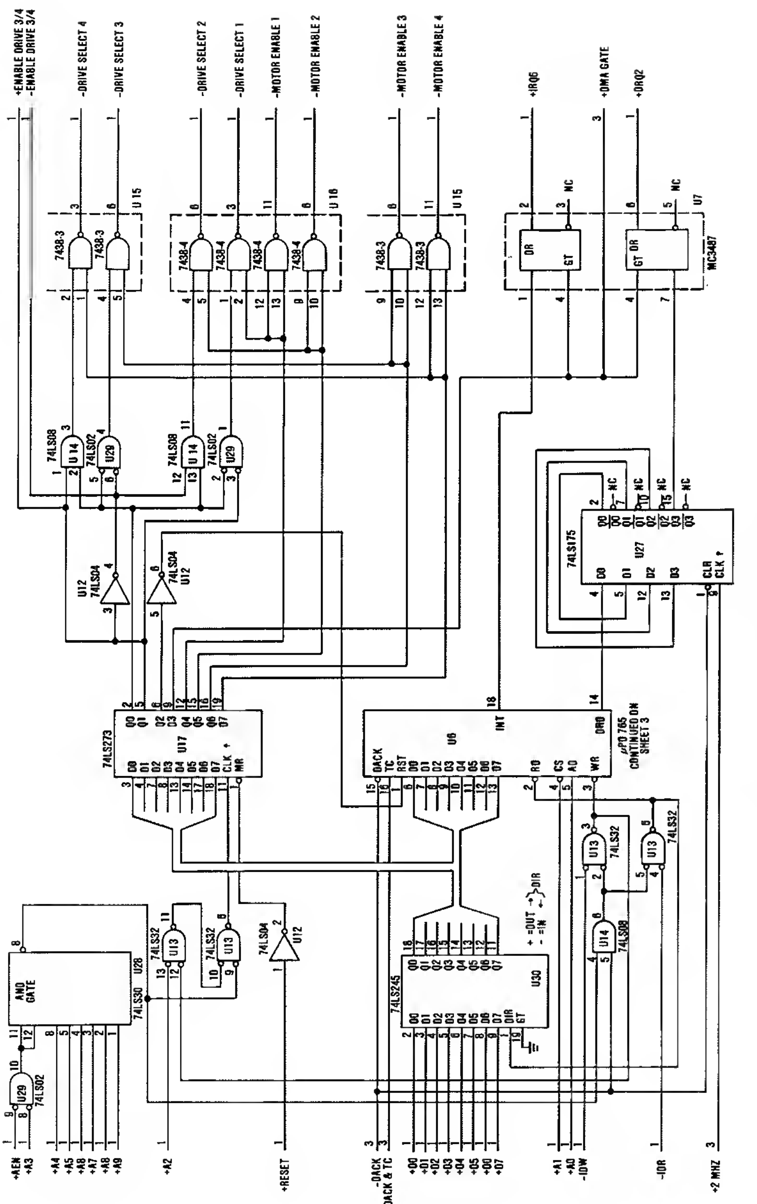
NOTES:

1. SIGNALS ON DRIVE PINS 10 THRU 16 ARE SWAPPED BY THE DRIVE CABLE BETWEEN DRIVES 1 & 2 (AND 3 & 4) AS FOLLOWS:

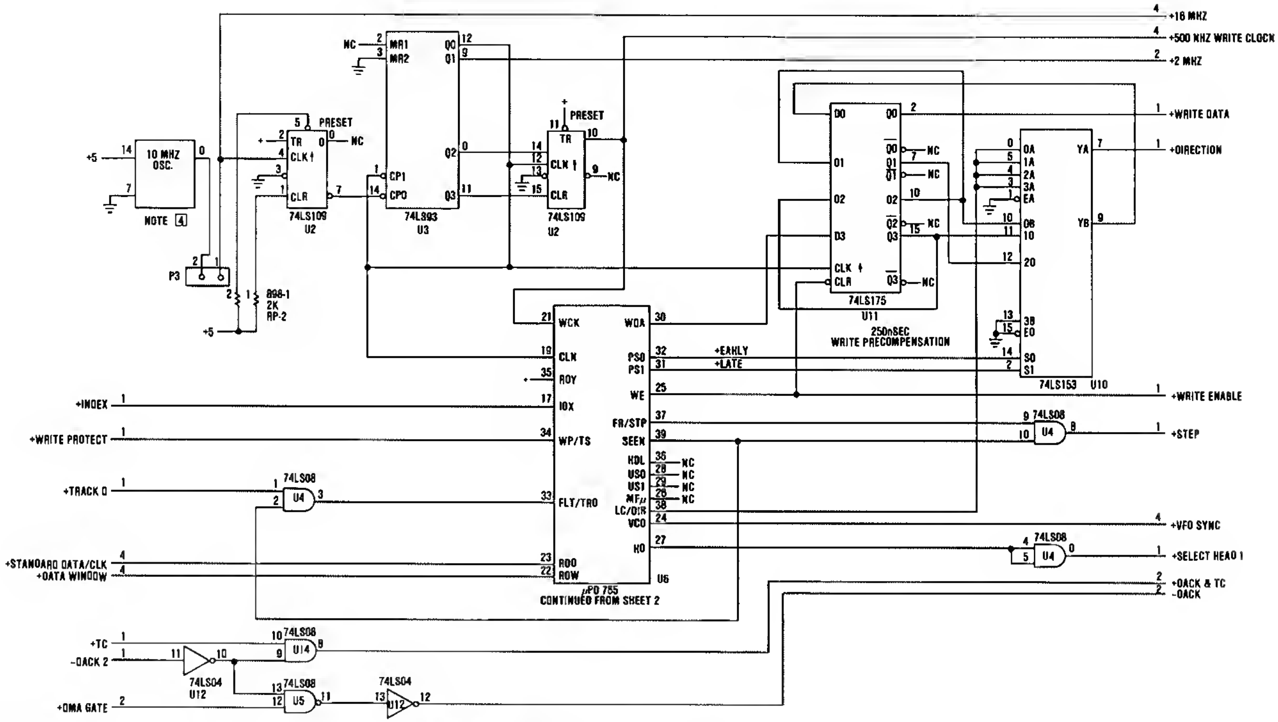
10 TO	16
11	15
12	14
13	13
14	12
15	11
16 TO	10

2. ALL DRIVES ARE JUMPERED FOR MULTIPLEX OPERATION. HEAD LOAD WITH DRIVE SELECT AND DRIVE SELECT VIA INPUT PIN 12. TERMINATING R-PACS ARE LEFT IN DRIVES 1 & 3 ONLY
3. .047µFD SHOULD BE ADJACENT TO MODULES MC3487, 7438, 74S74, 18 MHZ OSC, RP-1, MC4044, MC4024, 74LS181 & 74LS191 8.2µFD CAPS SHOULD BE NEAR ASSOCIATED P1 PINS.
4. ALL SIGNAL LINES HIGHER THAN OR EQUAL TO 1MHZ SHOULD BE KEPT TO THE SHORTEST POSSIBLE LENGTH. THIS IS A PRIMARY DESIGN GOAL
5. MAKE NO CONNECTION TO UNUSED PINS ON THE VCO, CHARGE PUMP & DATA SEPARATOR MODULES
6. ALL VOLTAGE AND GROUND CONNECTORS TO THE VCO, CHARGE PUMP AND ASSOCIATED DISCRETE COMPONENTS SHOULD BE SEPARATE FROM OTHER CIRCUITS AND THEN JOINED TO THE OTHER CIRCUITS AT ONE POINT

5-1/4 Inch Diskette Drive Adapter (Sheet 1 of 4)

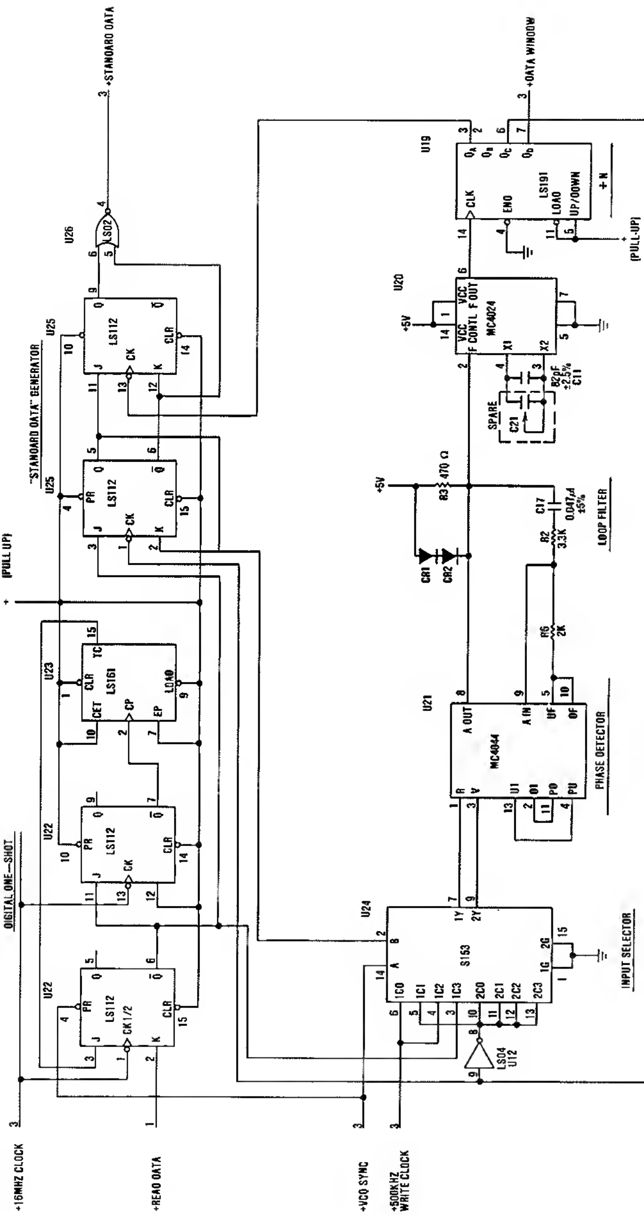


5-1/4 Inch Diskette Drive Adapter (Sheet 2 of 4)

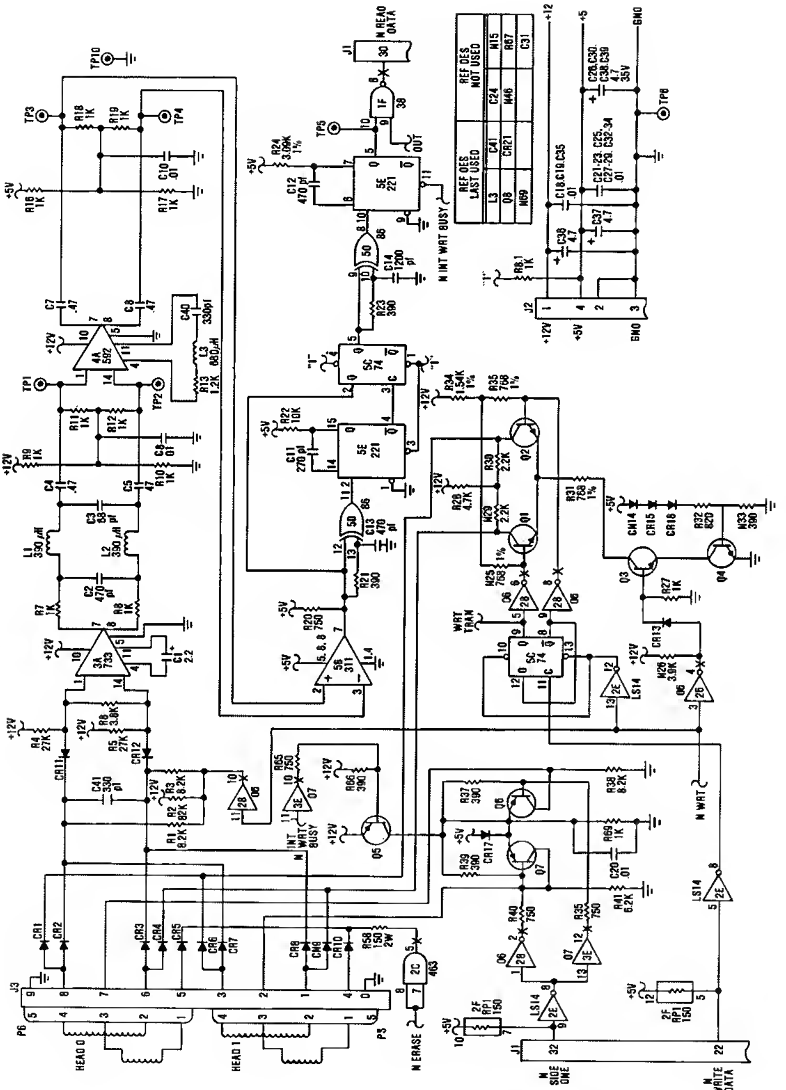


NOTE:
U4 (74LS08) PINS 12 AND 13 ARE CONNECTED ONLY ON CARDS BUILT
USING RAW CARD P/N 5001293

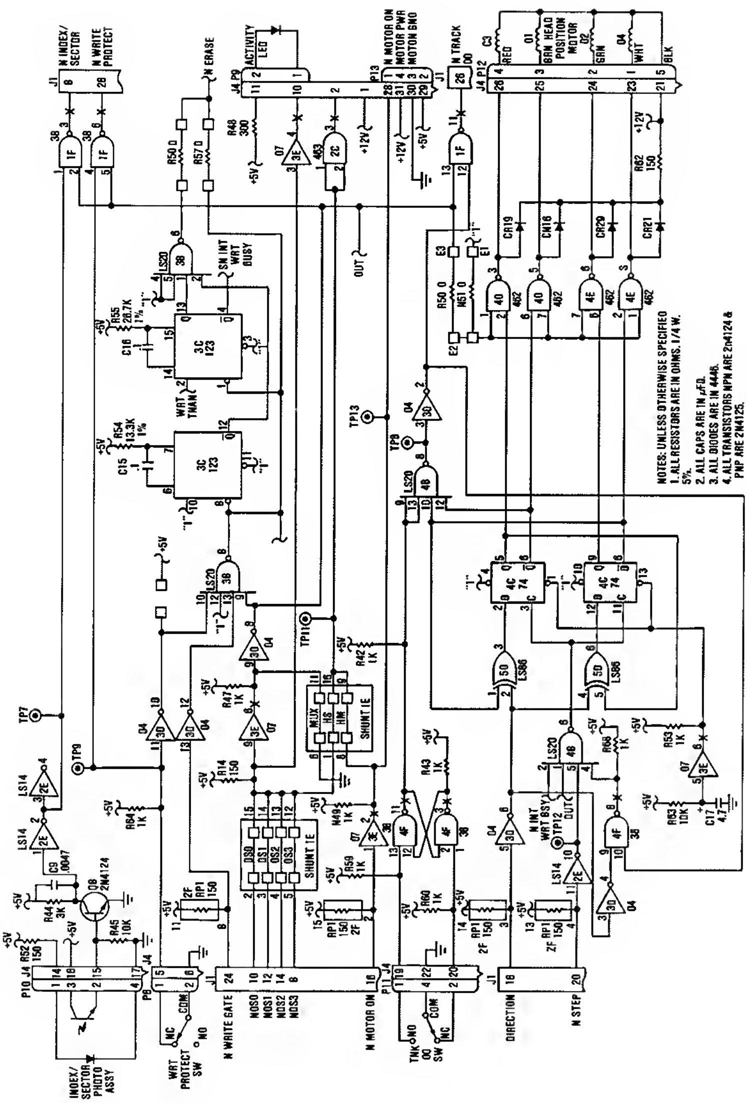
5-1/4 Inch Diskette Drive Adapter (Sheet 3 of 4)



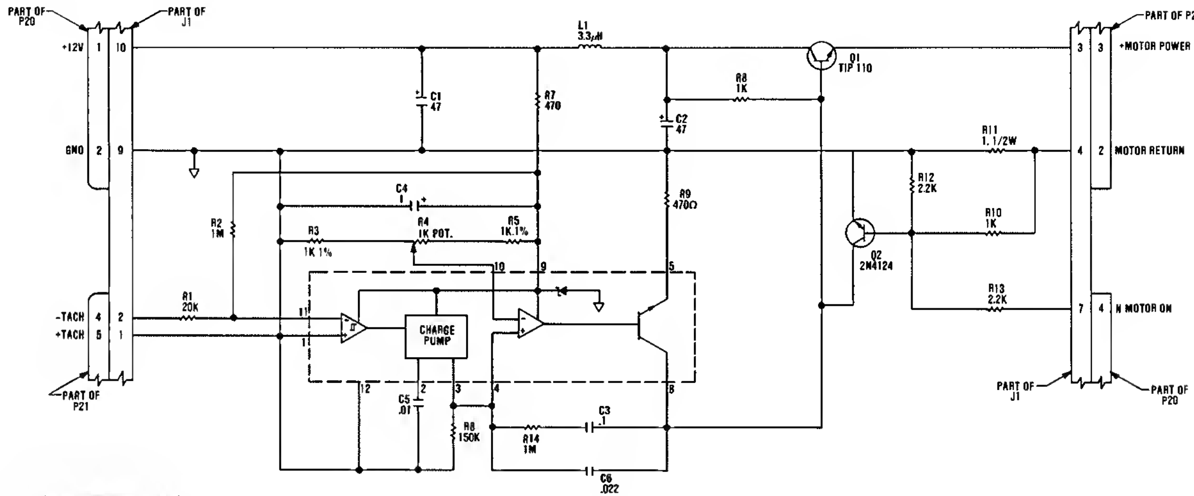
5-1/4 Inch Diskette Drive Adapter (Sheet 4 of 4)



5-1/4 Inch Diskette Drive Type 1 (Sheet 1 of 3)

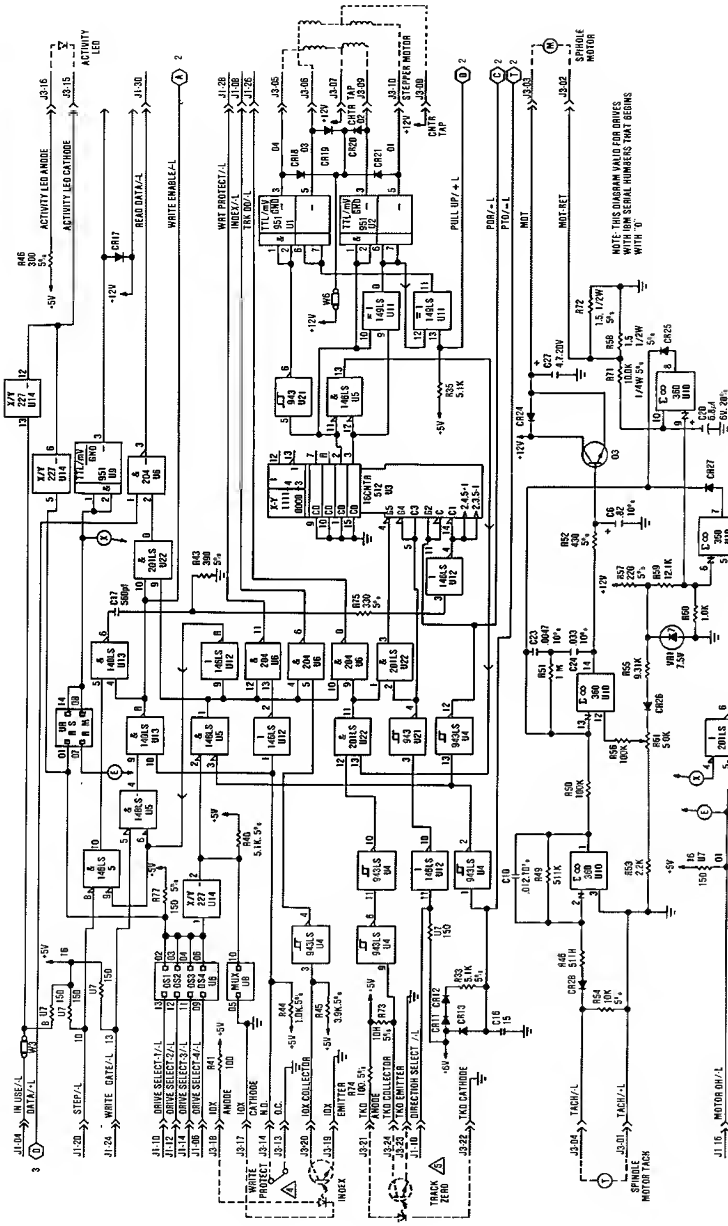


5-1/4 Inch Diskette Drive Type 1 (Sheet 2 of 3)

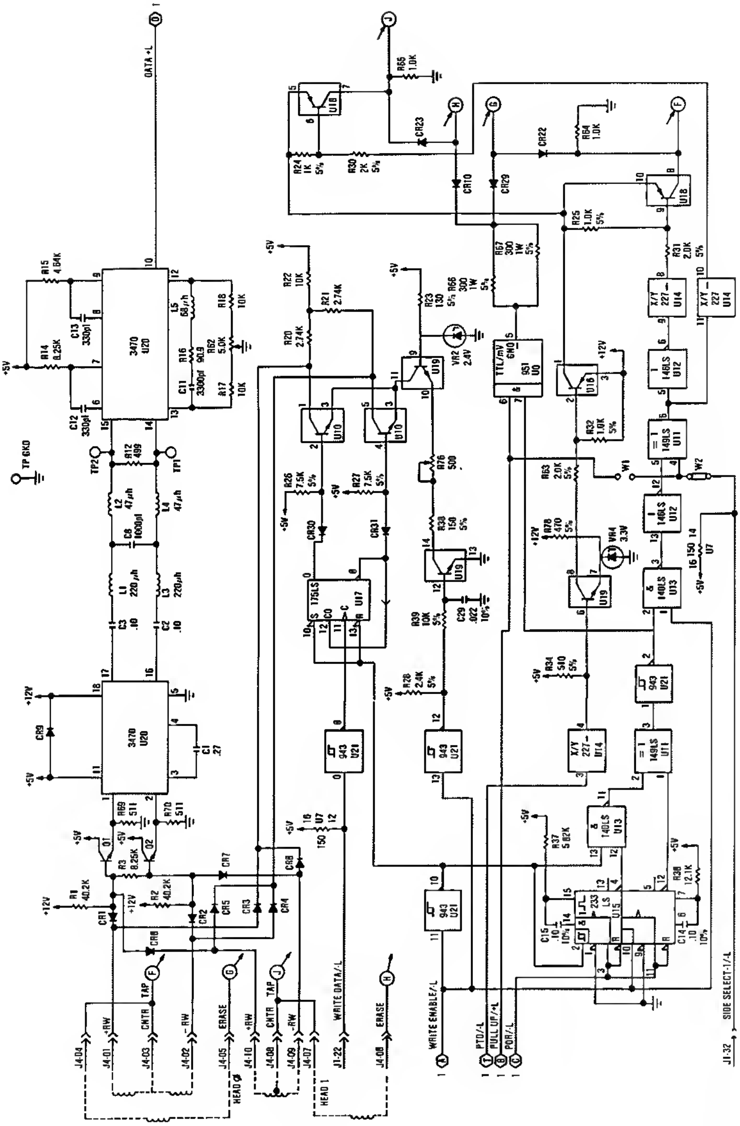


NOTES: UNLESS OTHERWISE SPECIFIED
1. RESISTORS ARE IN OHMS. +5%. 1/4W.
2. 1% RESISTORS ARE 1/8W.
3. CAPACITORS ARE IN μF. +20%. 35V.

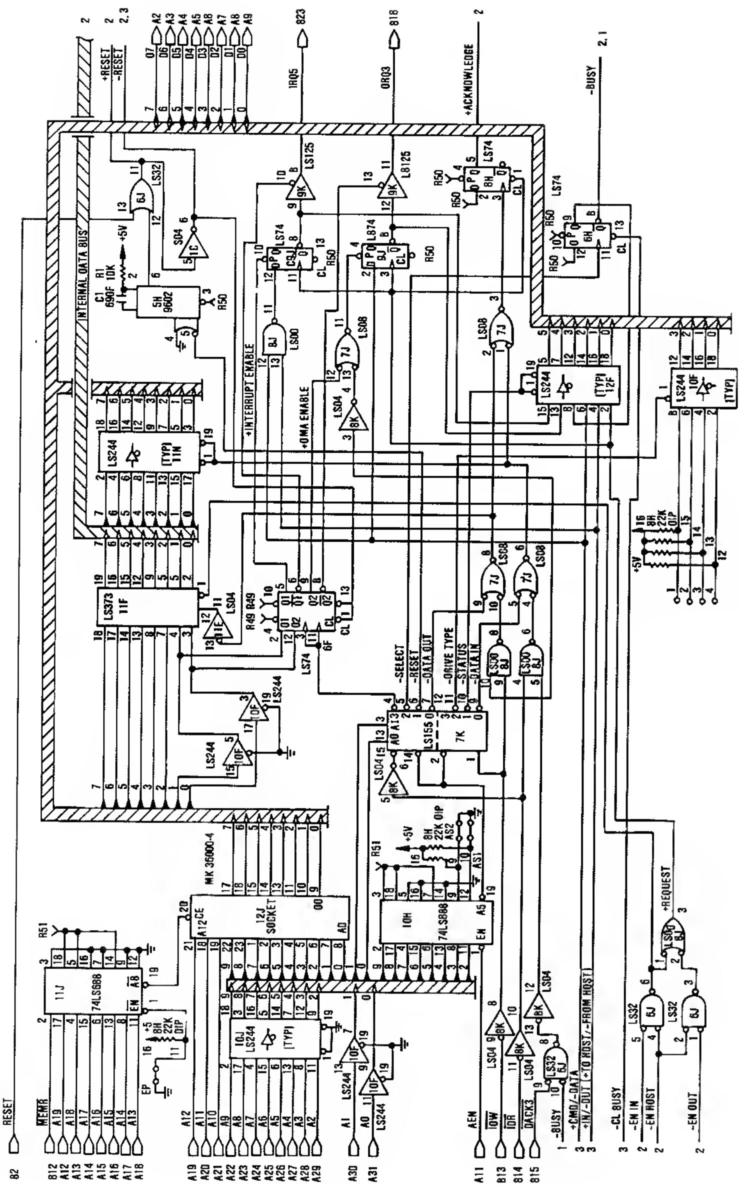
5-1/4 Inch Diskette Drive Type 1 (Sheet 3 of 3)



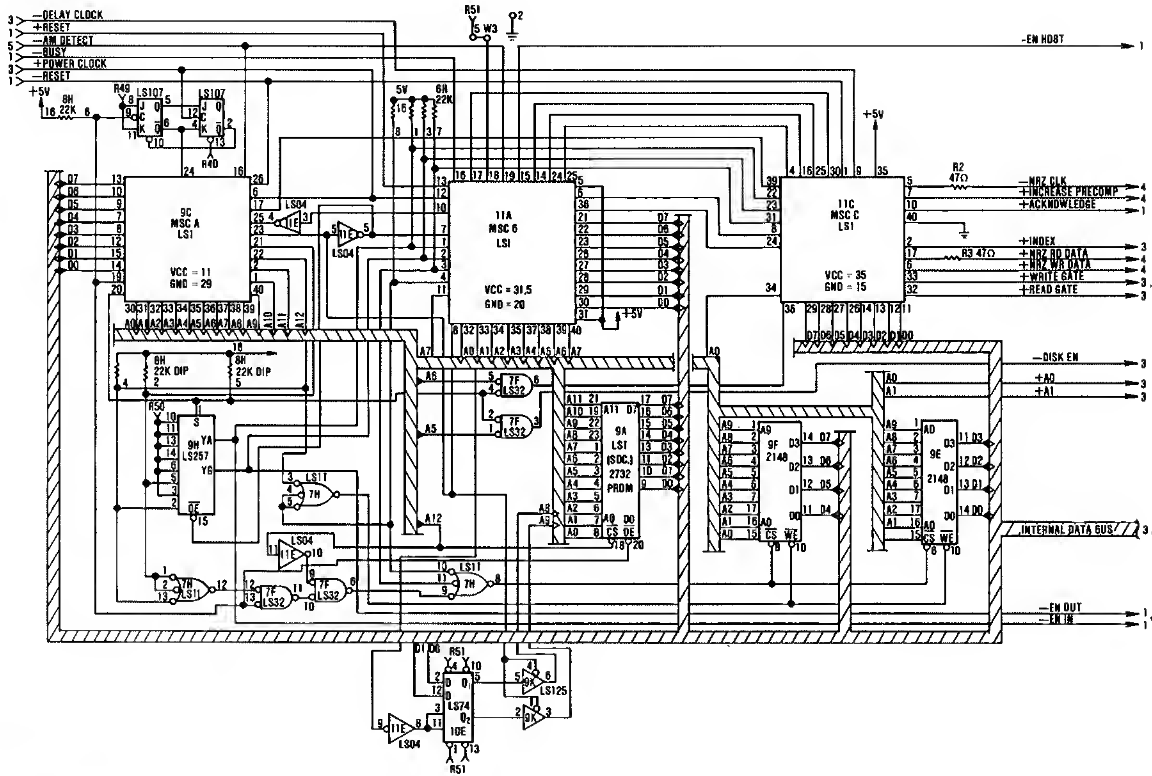
5-1/4 Inch Diskette Drive Type 2 (Sheet 1 of 2)



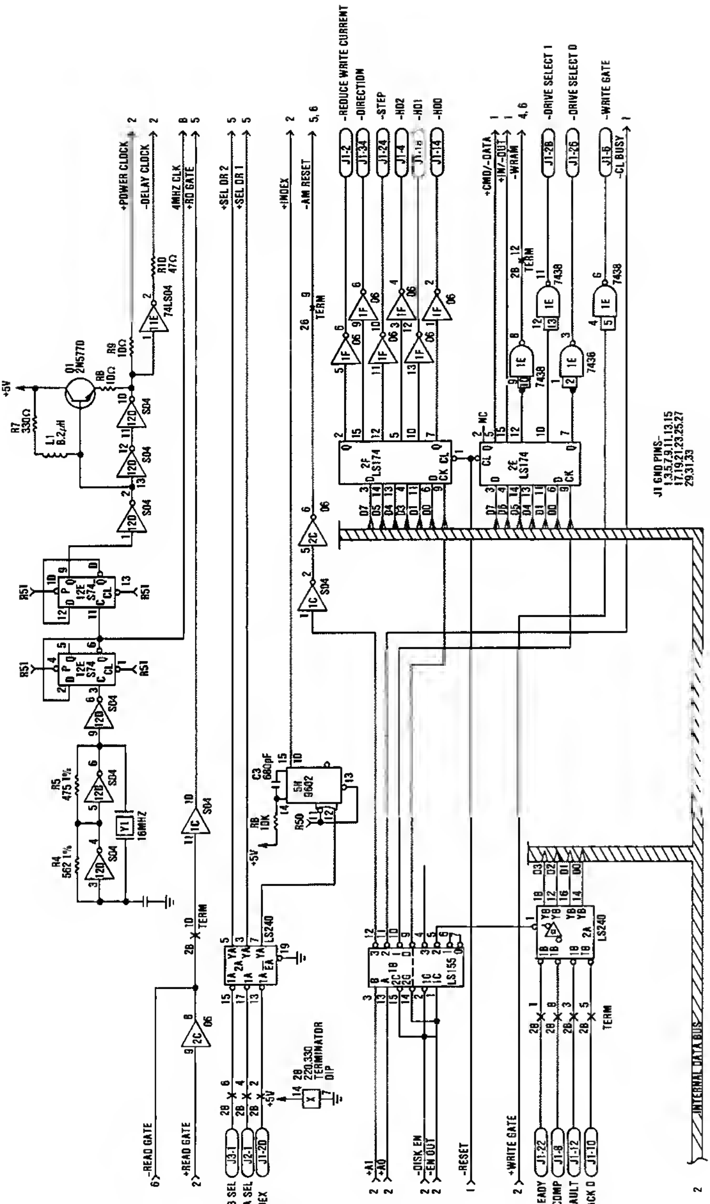
5-1/4 Inch Diskette Drive Type 2 (Sheet 2 of 2)



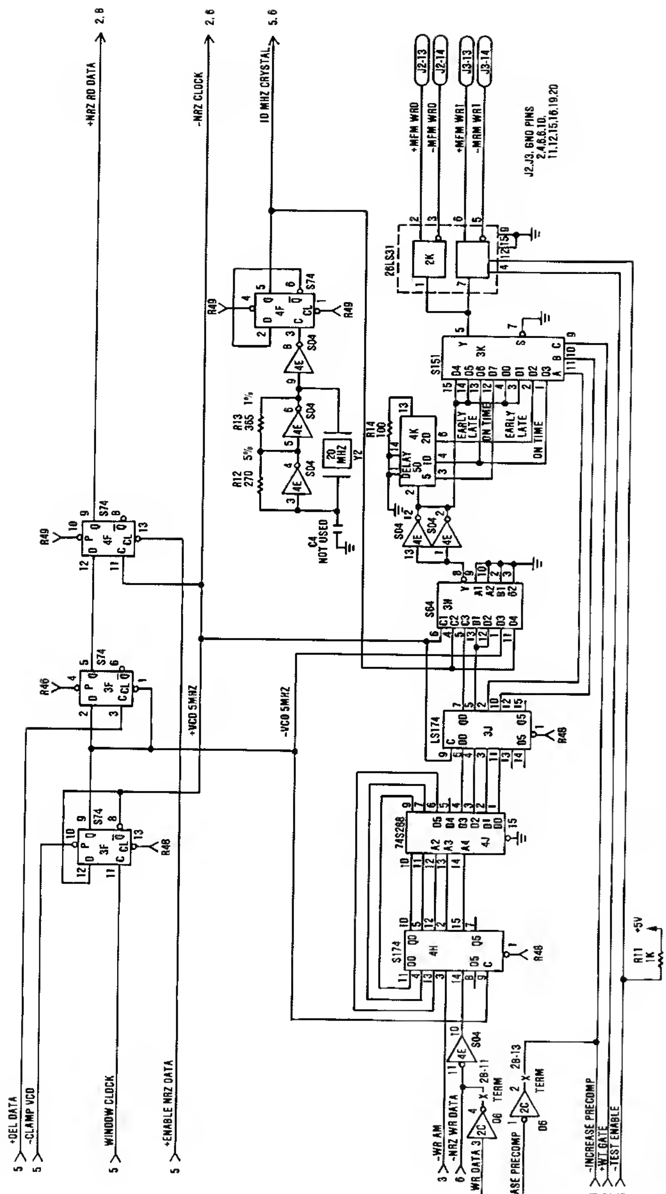
Fixed Disk Drive Adapter (Sheet 1 of 6)



Fixed Disk Drive Adapter (Sheet 2 of 6)

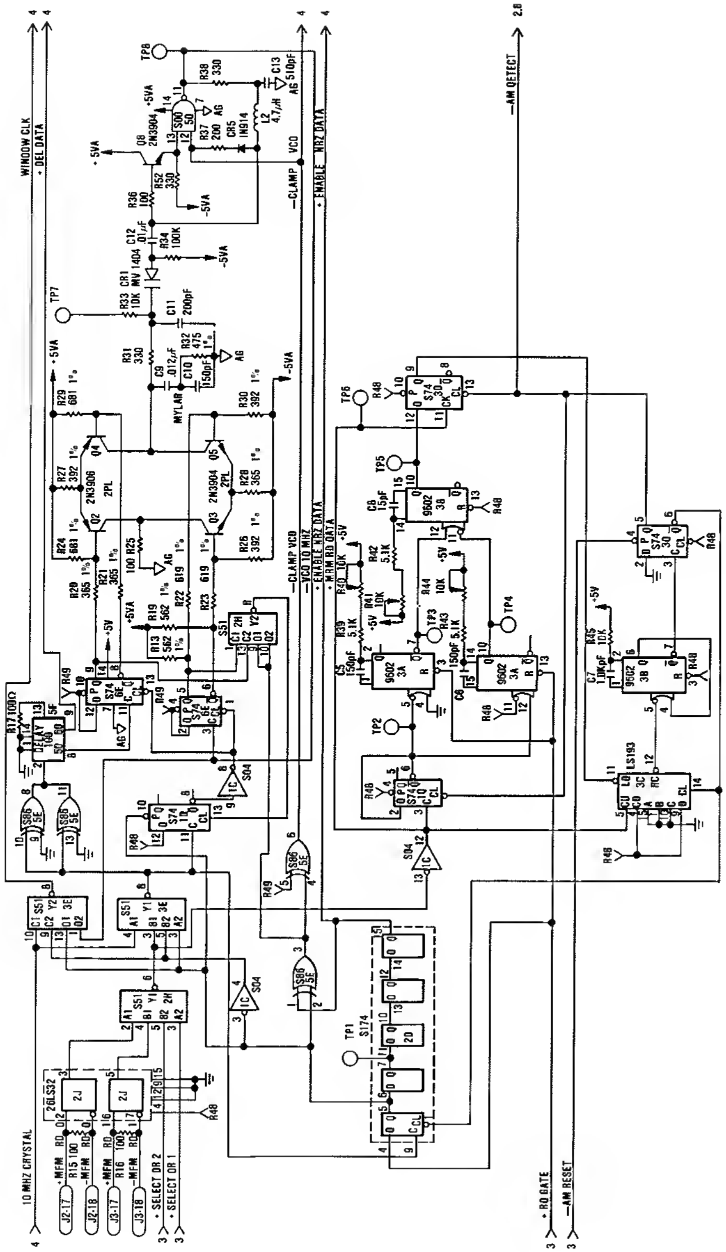


Fixed Disk Drive Adapter (Sheet 3 of 6)

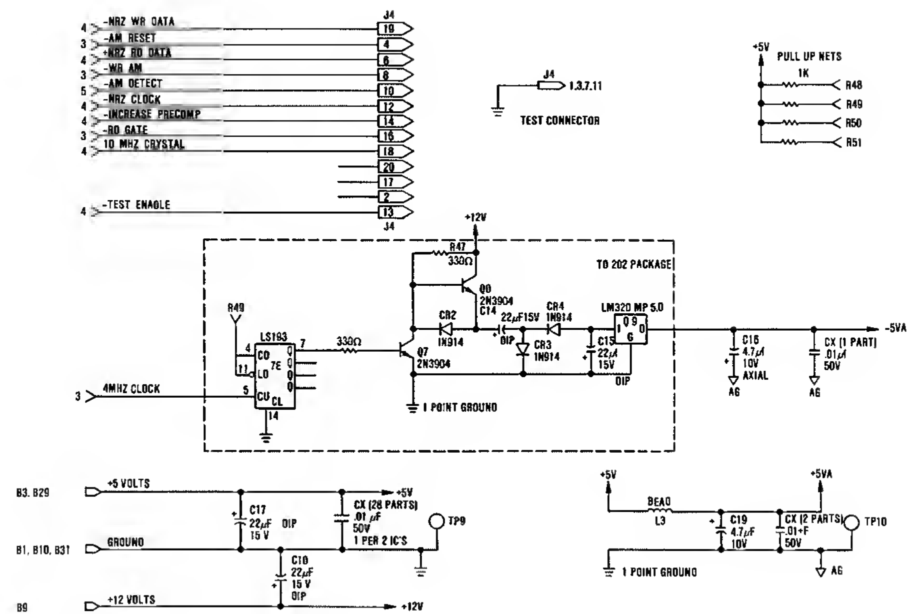


Fixed Disk Drive Adapter (Sheet 4 of 6)

Appendix D

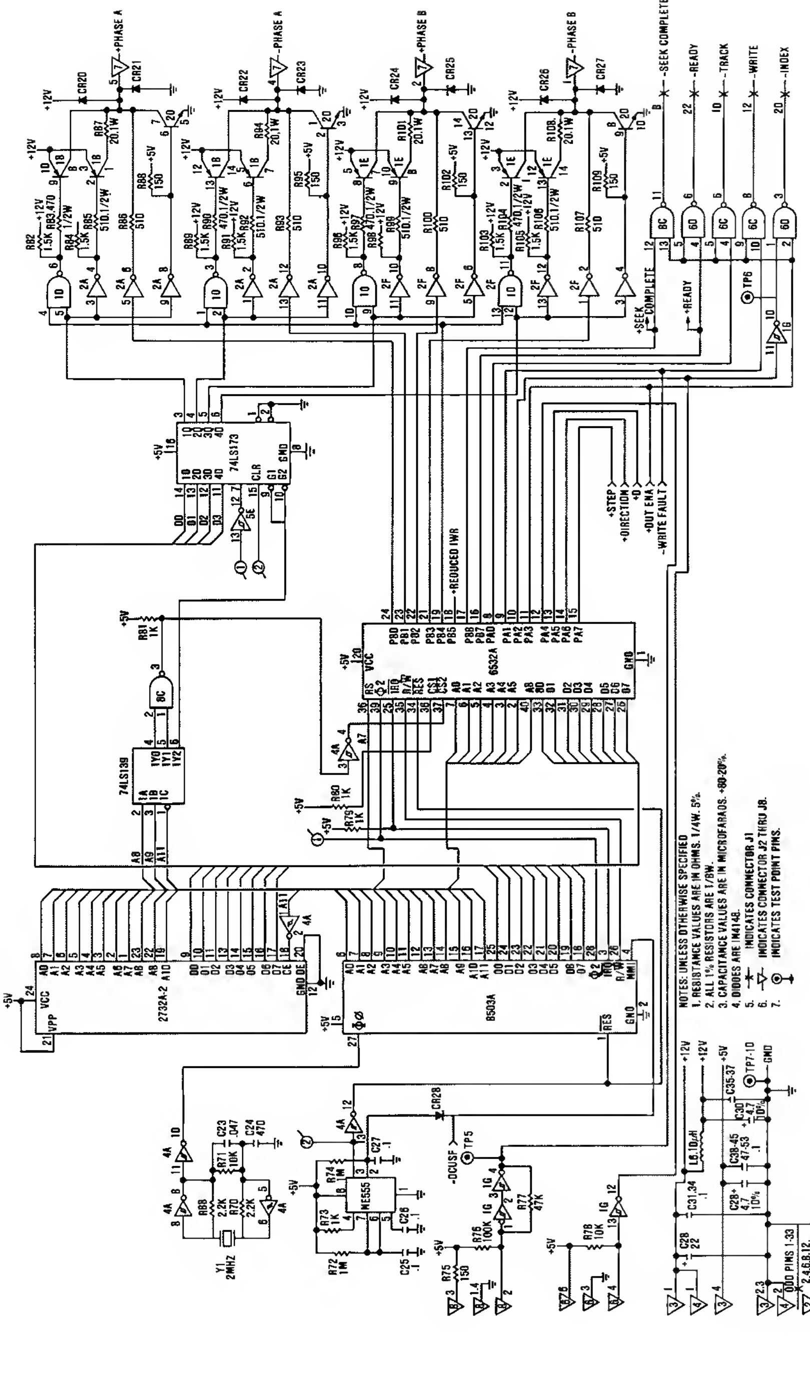


Fixed Disk Drive Adapter (Sheet 5 of 6)



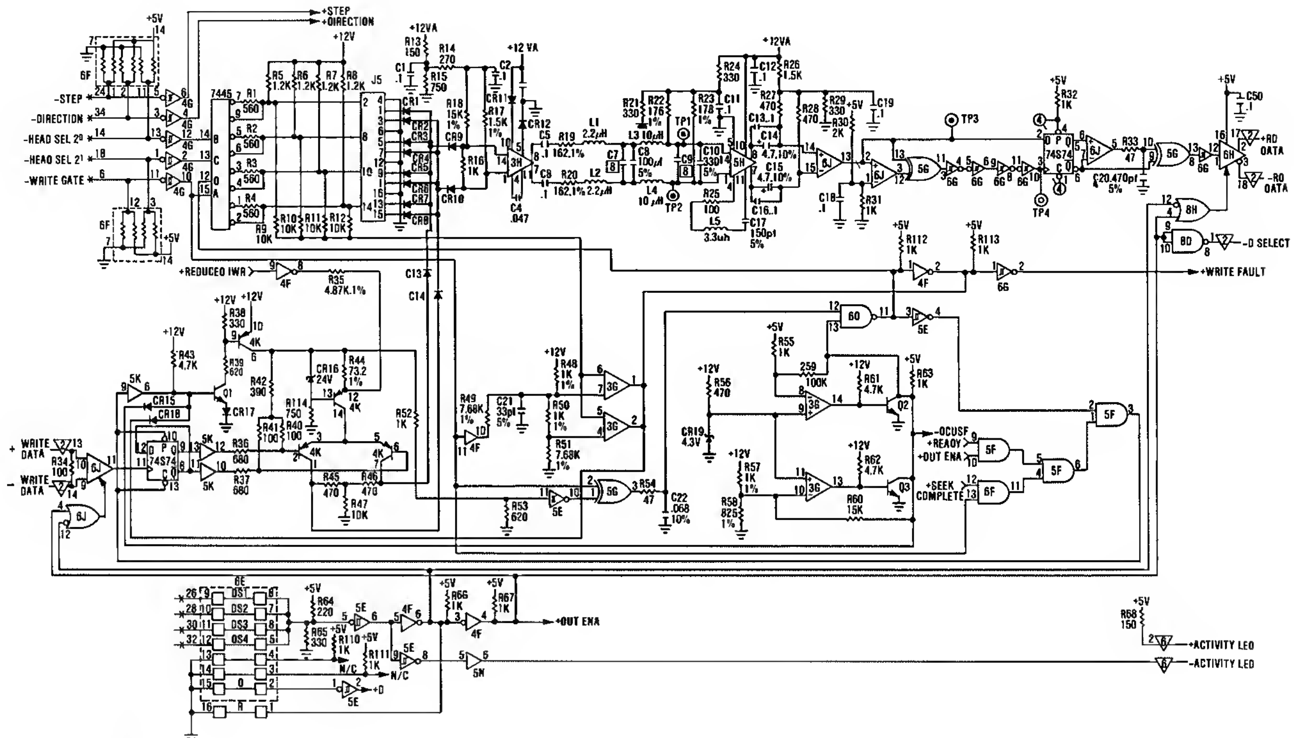
NOTES:
UNLESS OTHERWISE SPECIFIED:
1. ALL RESISTORS 1/4 W. 5%. CARBON FILTER.
2. ALL CAPS +10V OR GREATER +10%.
3. NO MORE THAN 15 LOADS PER PULLUP NET.

Fixed Disk Drive Adapter (Sheet 6 of 6)

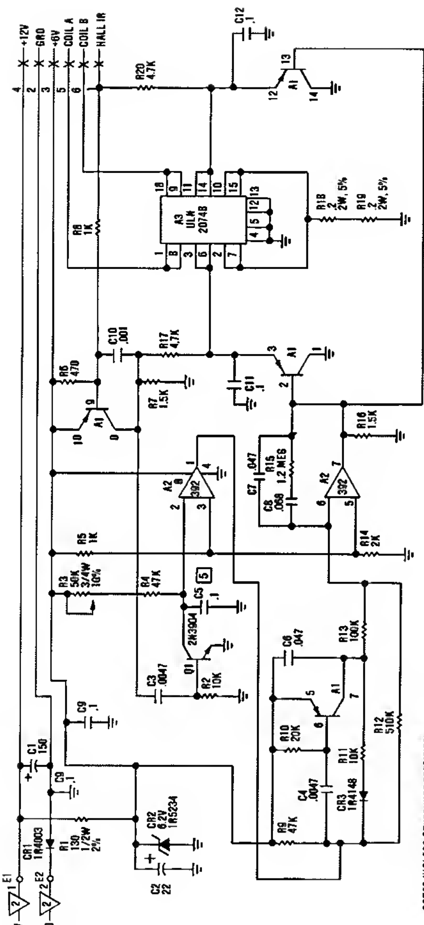


NOTES: UNLESS OTHERWISE SPECIFIED
1. RESISTANCE VALUES ARE IN OHMS. 1/4W. 5%.
2. ALL 1% RESISTORS ARE 1/8W.
3. CAPACITANCE VALUES ARE IN MICROFARADS. +80-20%.
4. DIODES ARE 1N4148.
5. INDICATES CONNECTOR J1
6. INDICATES CONNECTOR J2 THRU J8.
7. INDICATES TEST POINT PINS.

Fixed Disk Drive - Type 1 (Sheet 1 of 3)

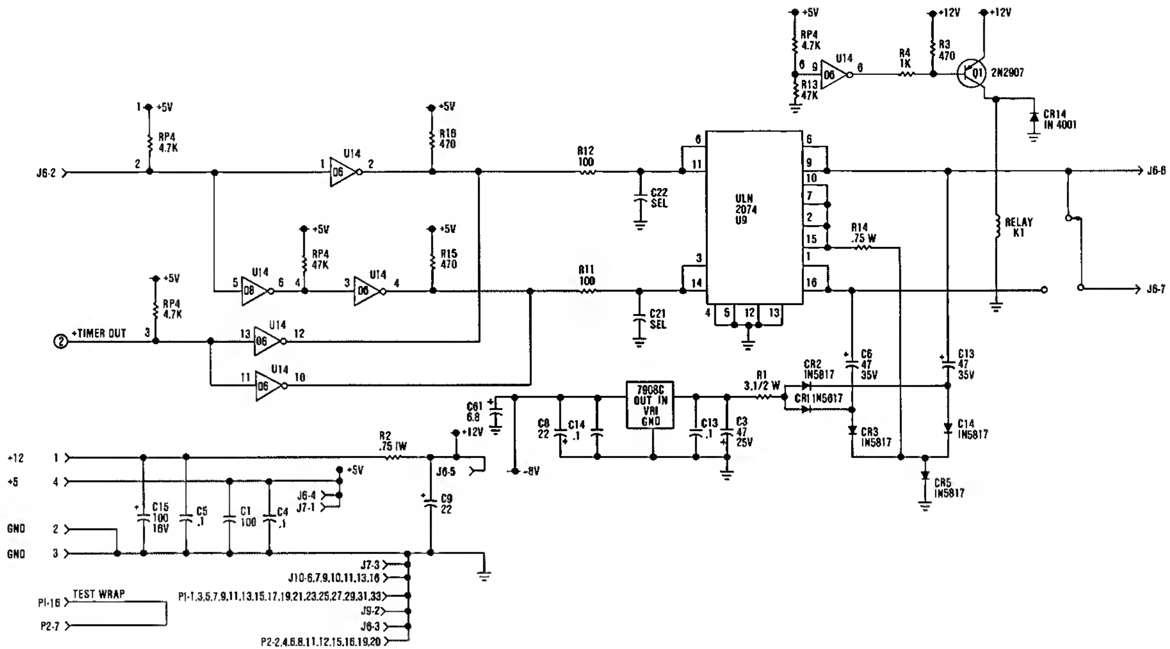


Fixed Disk Drive - Type 1 (Sheet 2 of 3)



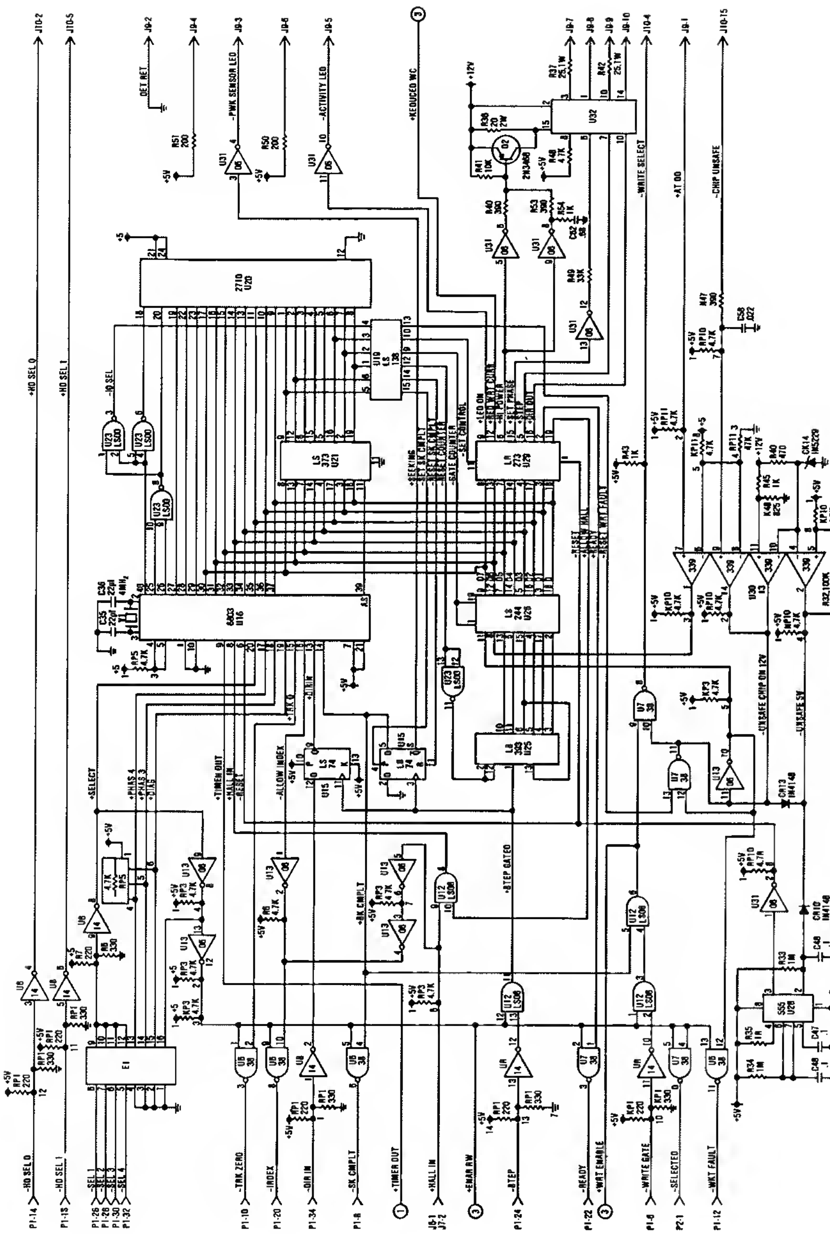
NOTES (UNLESS OTHERWISE SPECIFIED):
1. ALL RESISTORS ARE IN OHMS 1/4W, 5%.
2. ALL CAPACITORS ARE IN MICROFARADS, 10%.
3. —✕— INDICATES J1.
4. [2] INDICATES J2.
[5] C5, POLYCARBONATE 50V, 10%.

Fixed Disk Drive - Type 1 (Sheet 3 of 3)

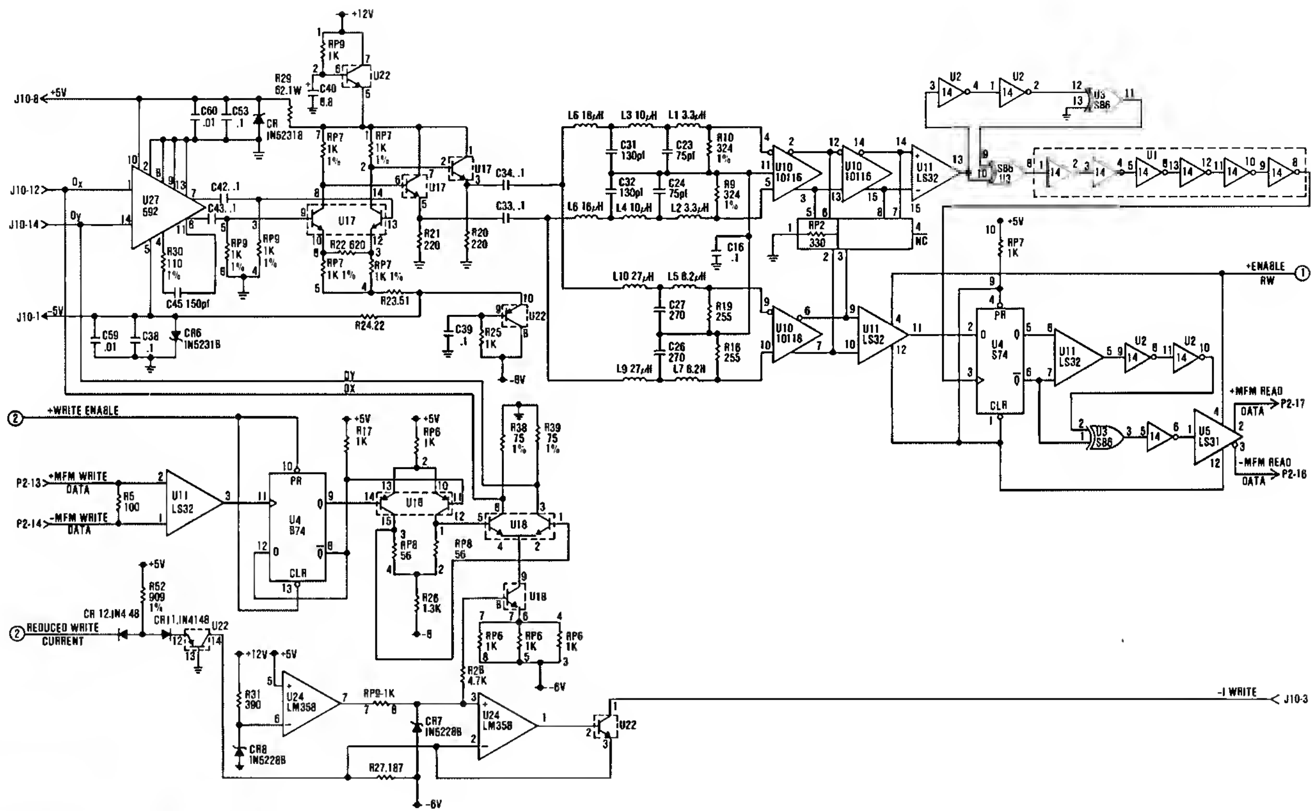


NOTES:
1. SHEET TO SHEET CONNECTION IS AS FOLLOWS:
SHEET NO.
2. UNLESS OTHERWISE SPECIFIED ALL RESISTORS ARE 1/4W 5% VALUE IN OHMS
CAPACITORS +80-20% VALUE IN μf
3. E1 IS A PROGRAMMABLE JUMPER SOCKET

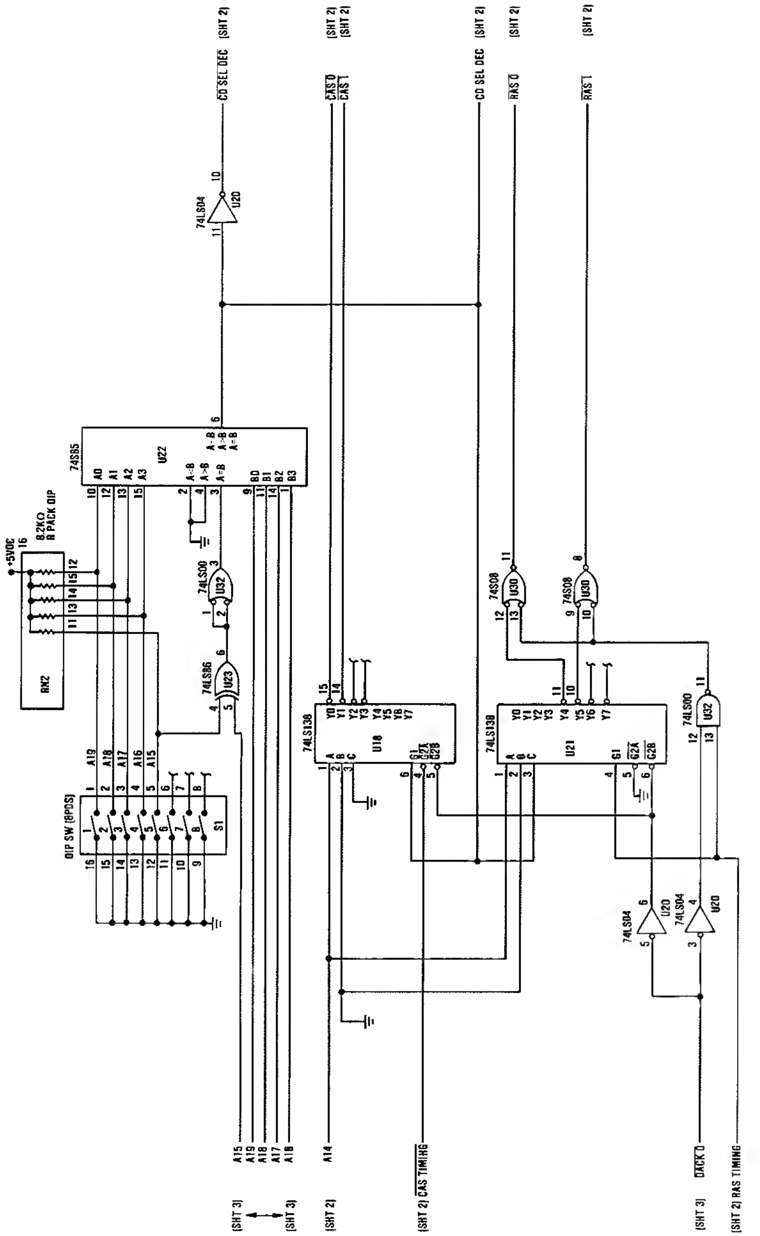
Fixed Disk Drive - Type 2 (Sheet 1 of 3)



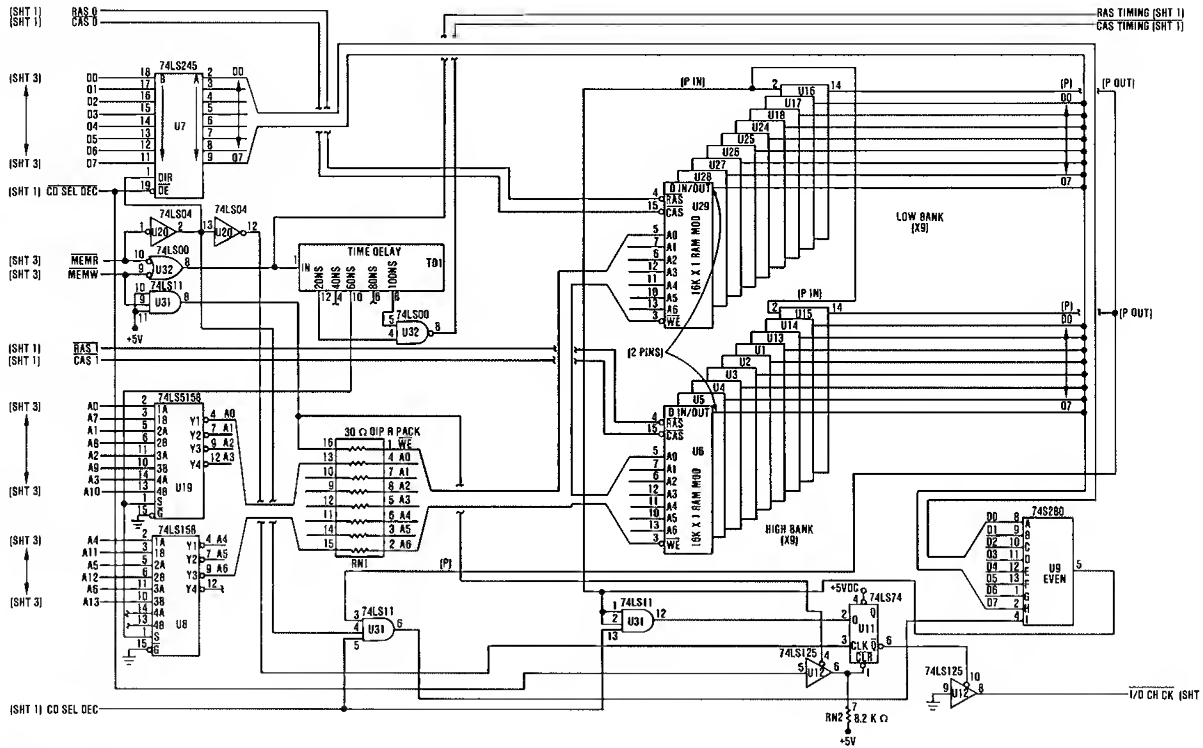
Fixed Disk Drive - Type 2 (Sheet 2 of 3)



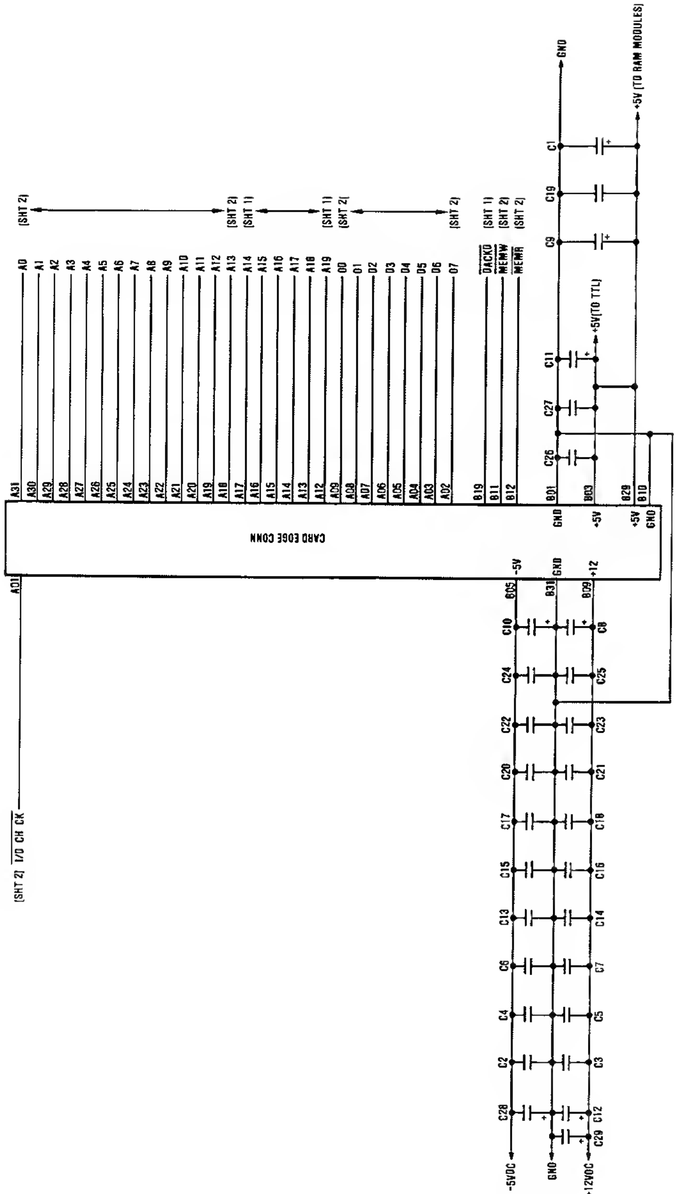
Fixed Disk Drive - Type 2 (Sheet 3 of 3)



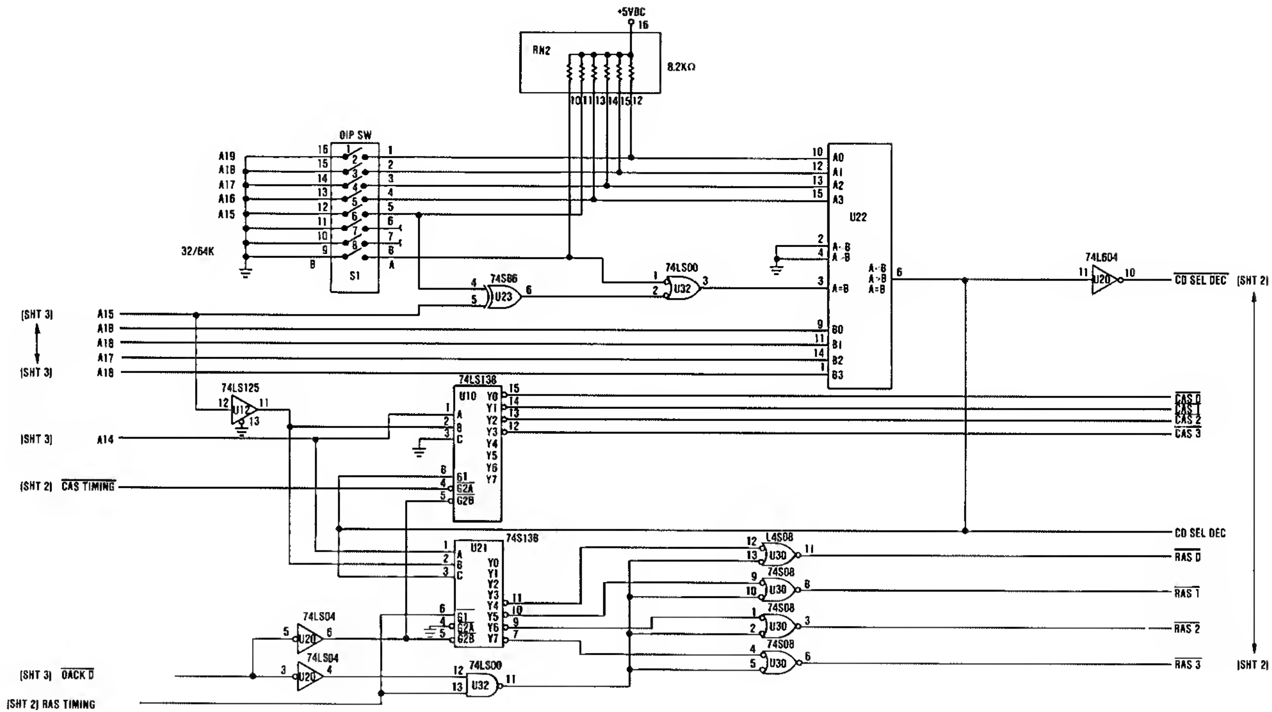
32K Memory Expansion Option (Sheet 1 of 3)



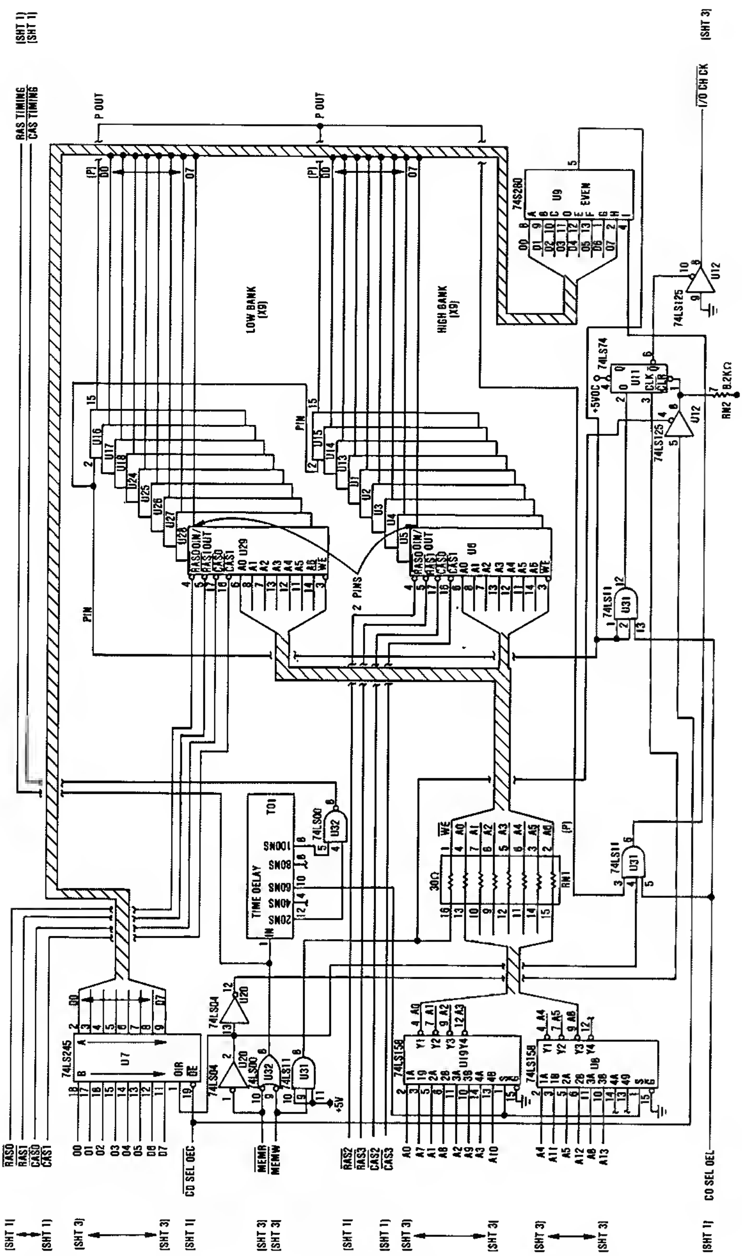
32K Memory Expansion Option (Sheet 2 of 3)



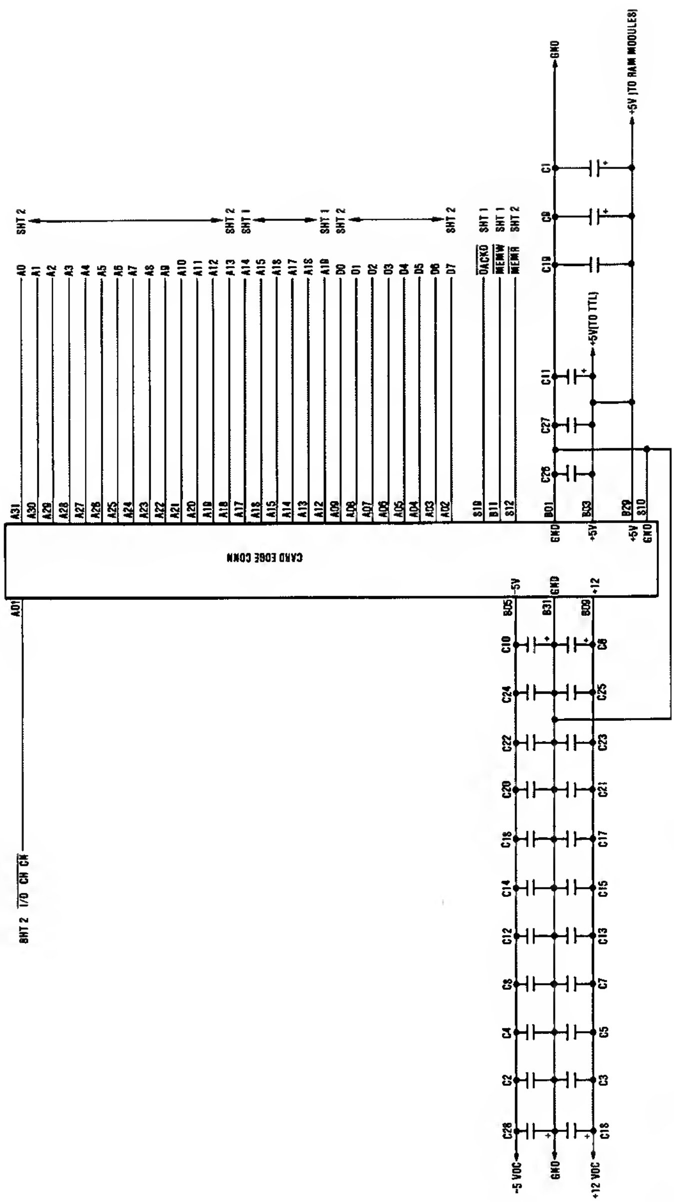
32K Memory Expansion Option (Sheet 3 of 3)



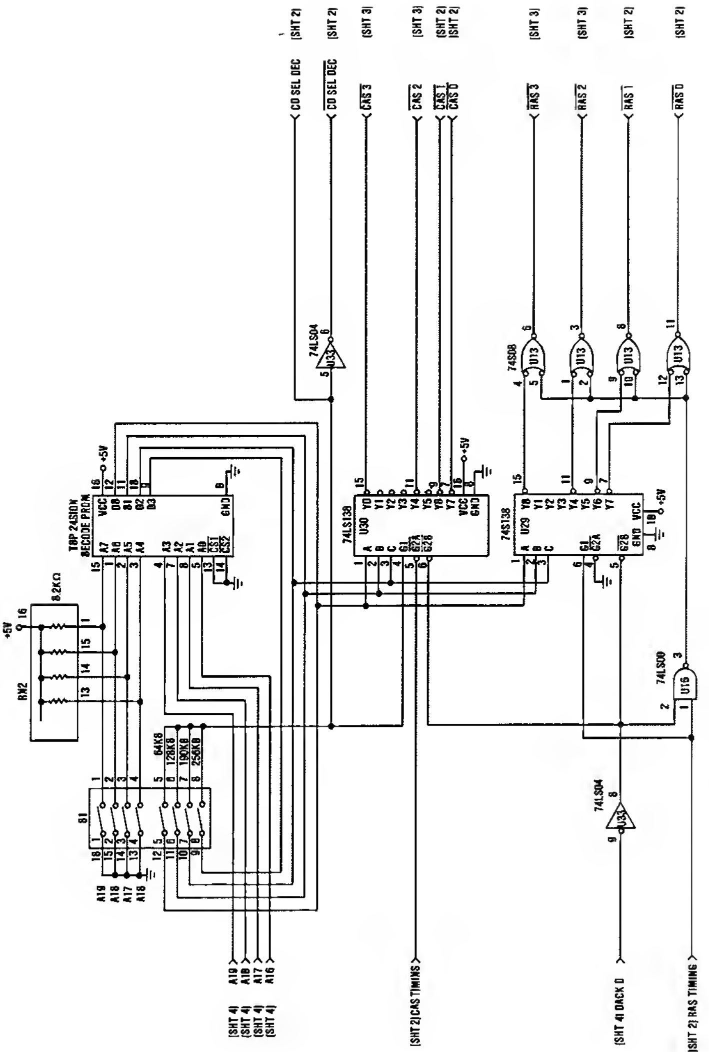
64K Memory Expansion Option (Sheet 1 of 3)



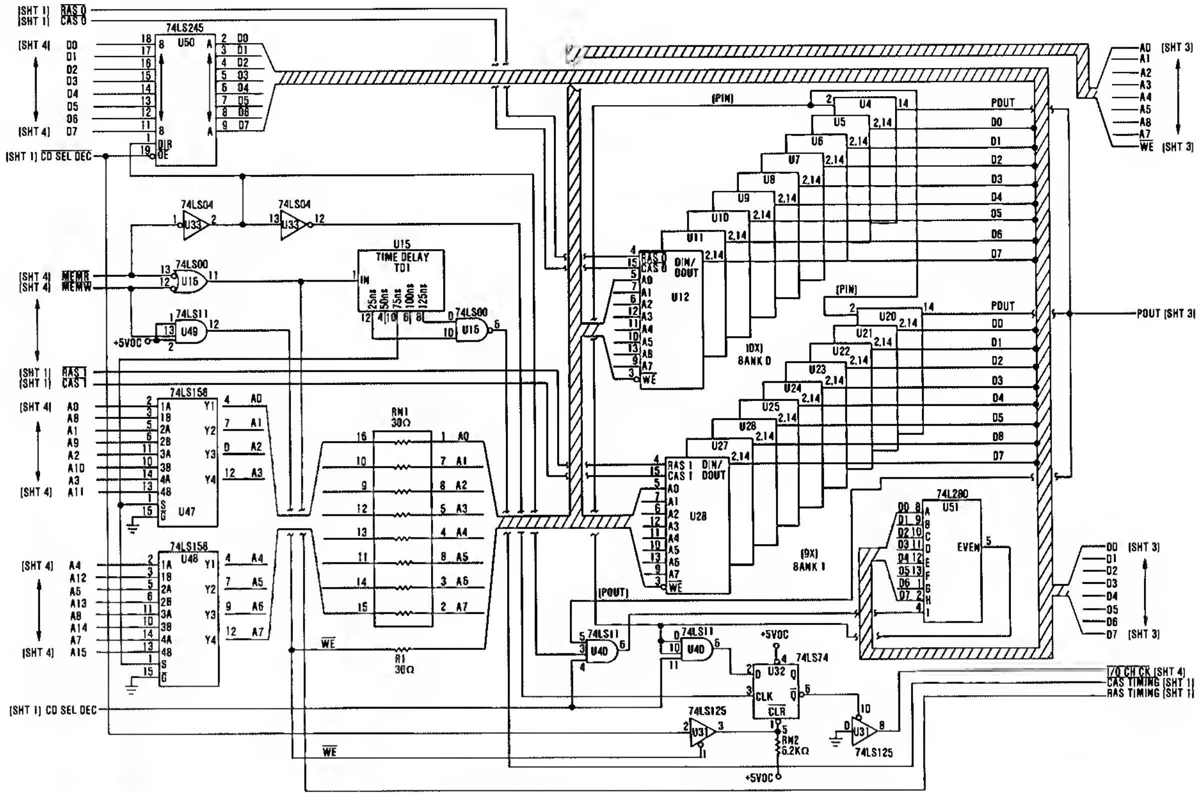
64K Memory Expansion Option (Sheet 2 of 3)



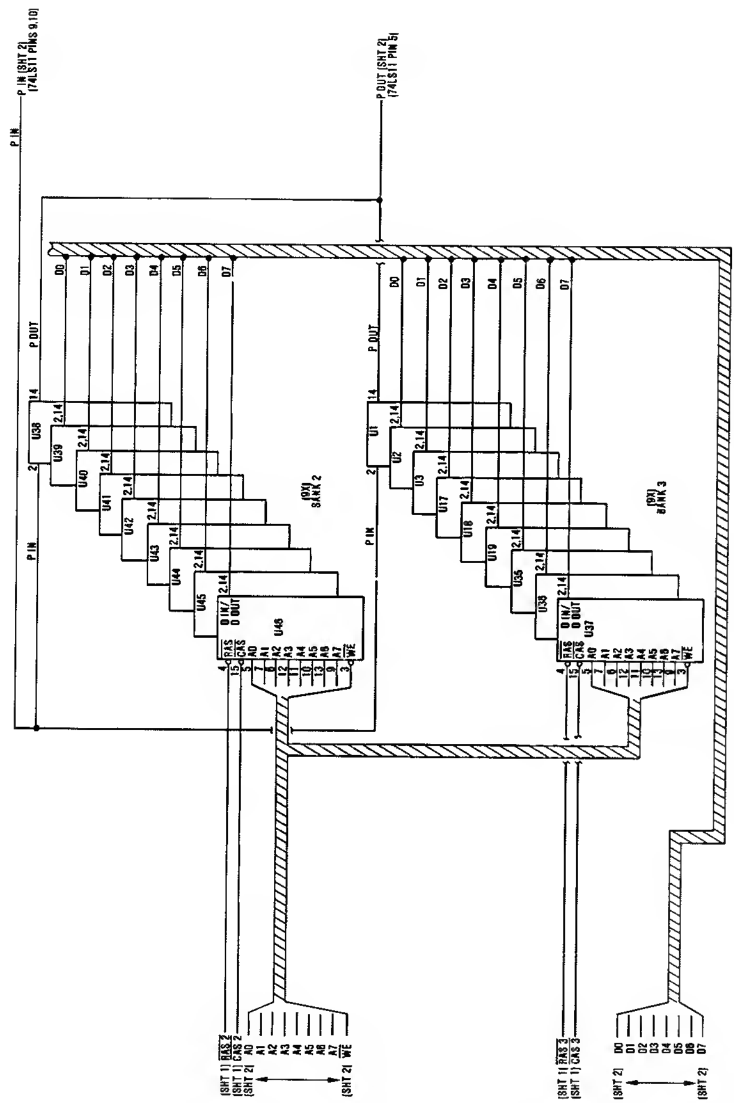
64K Memory Expansion Option (Sheet 3 of 3)



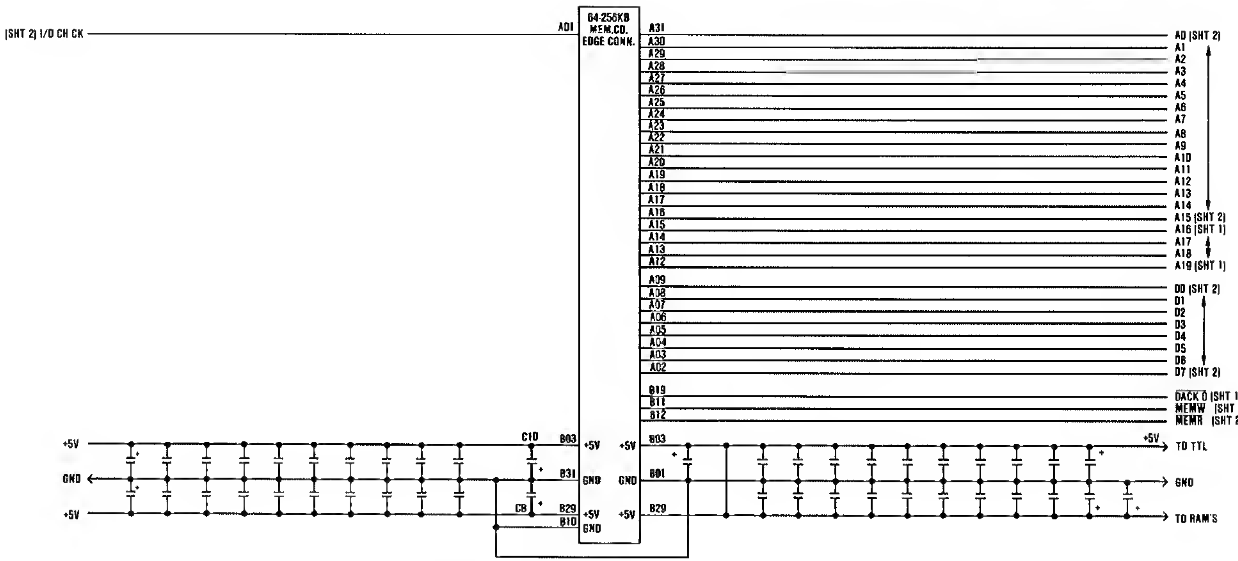
64/256K Memory Expansion Option (Sheet 1 of 4)



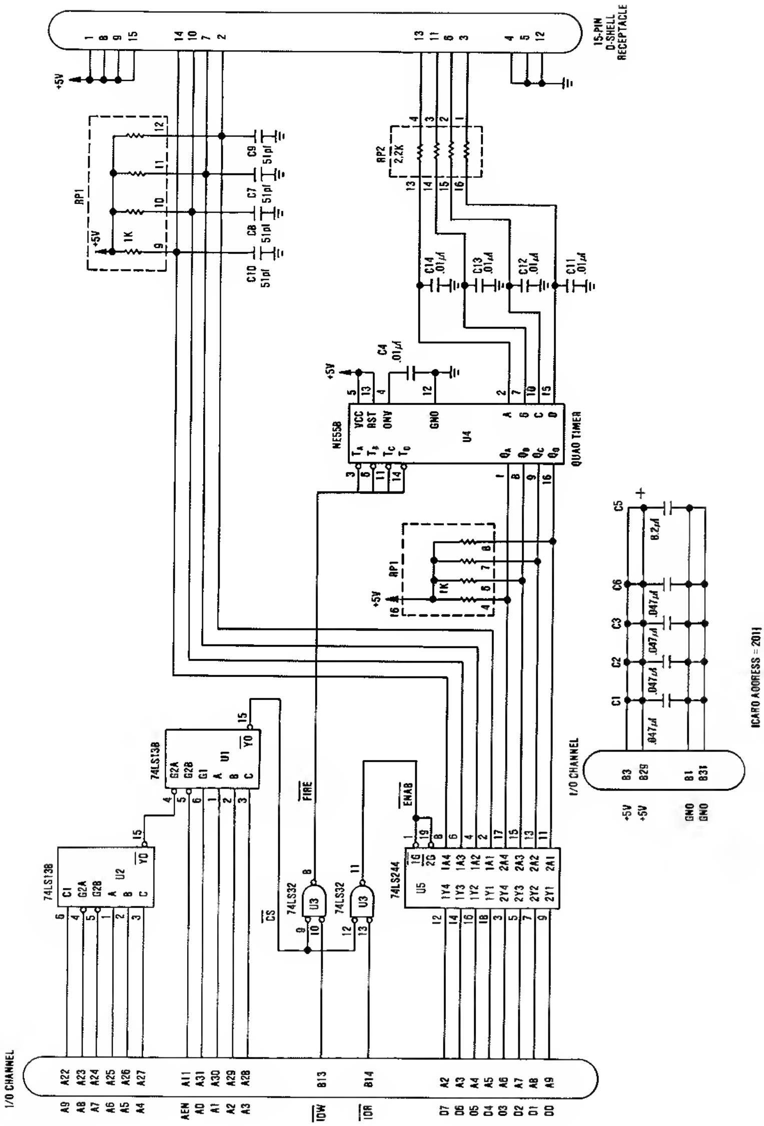
64/256K Memory Expansion Option (Sheet 2 of 4)



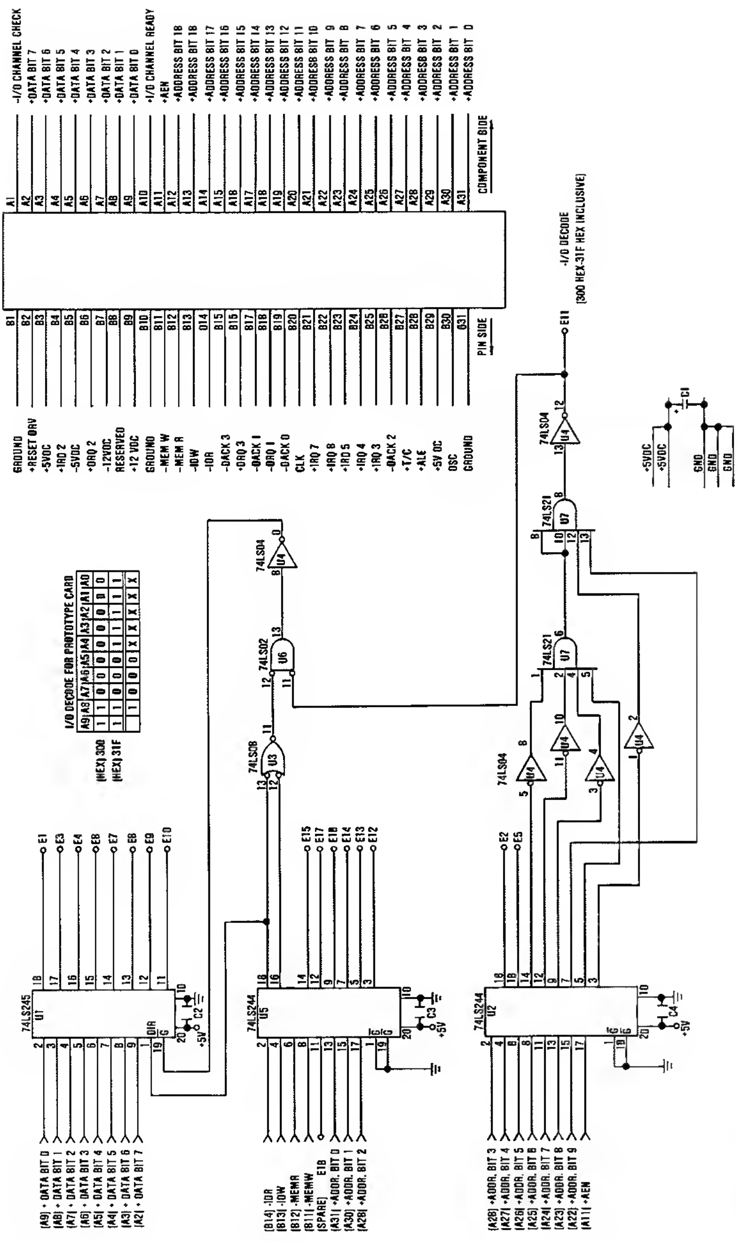
64/256K Memory Expansion Option (Sheet 3 of 4)



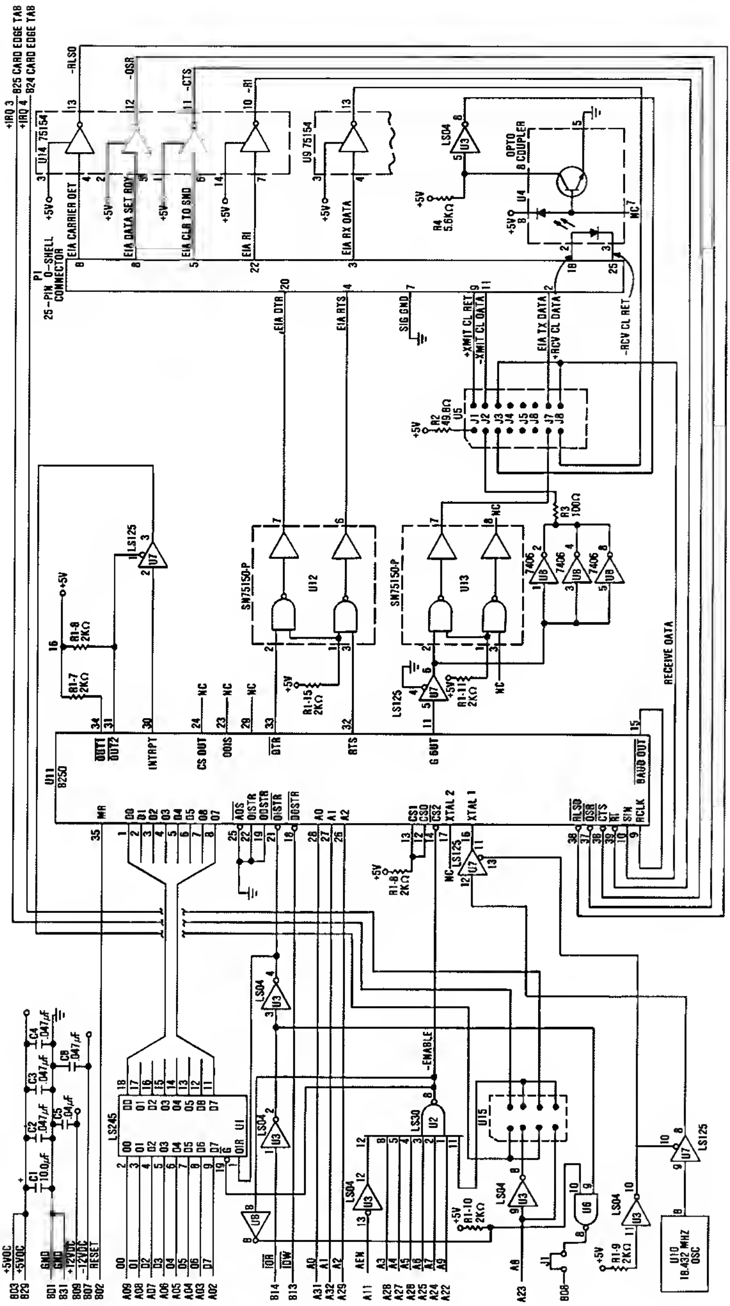
64/256K Memory Expansion Option (Sheet 4 of 4)



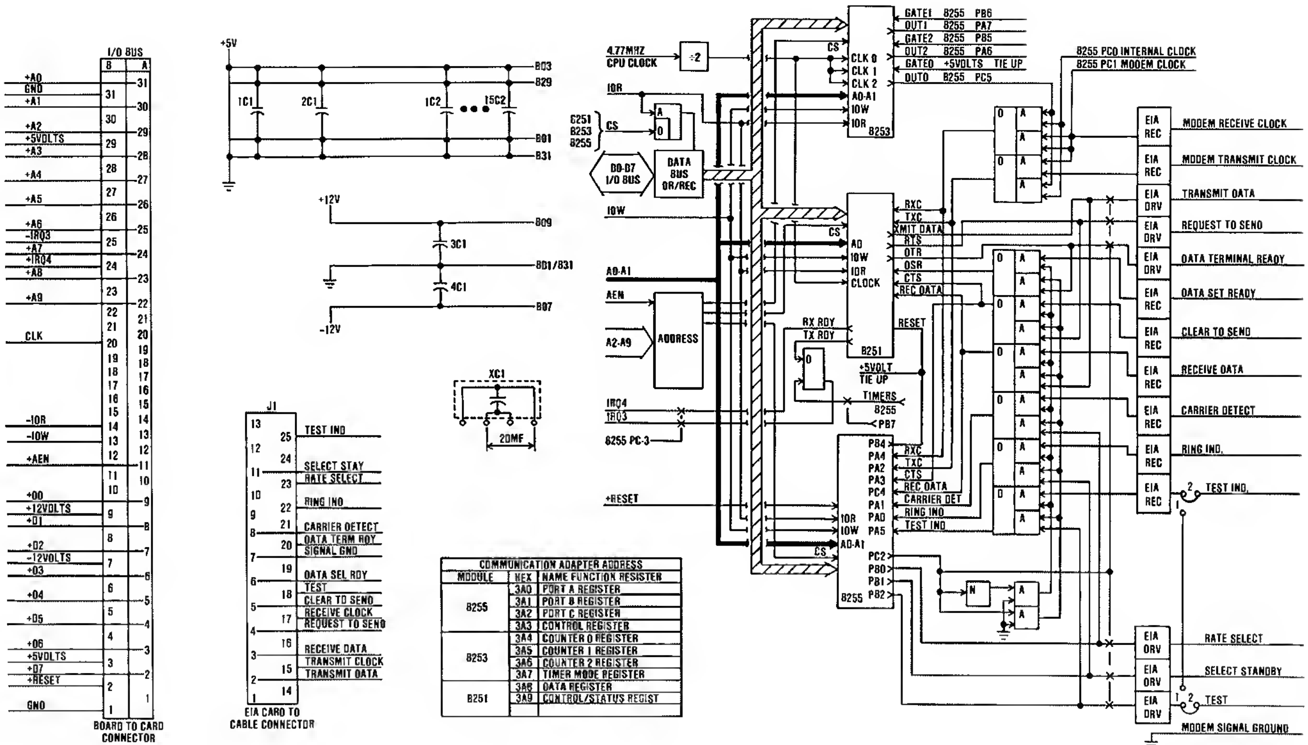
Game Control Adapter (Sheet 1 of 1)



Prototype Card (Sheet 1 of 1)

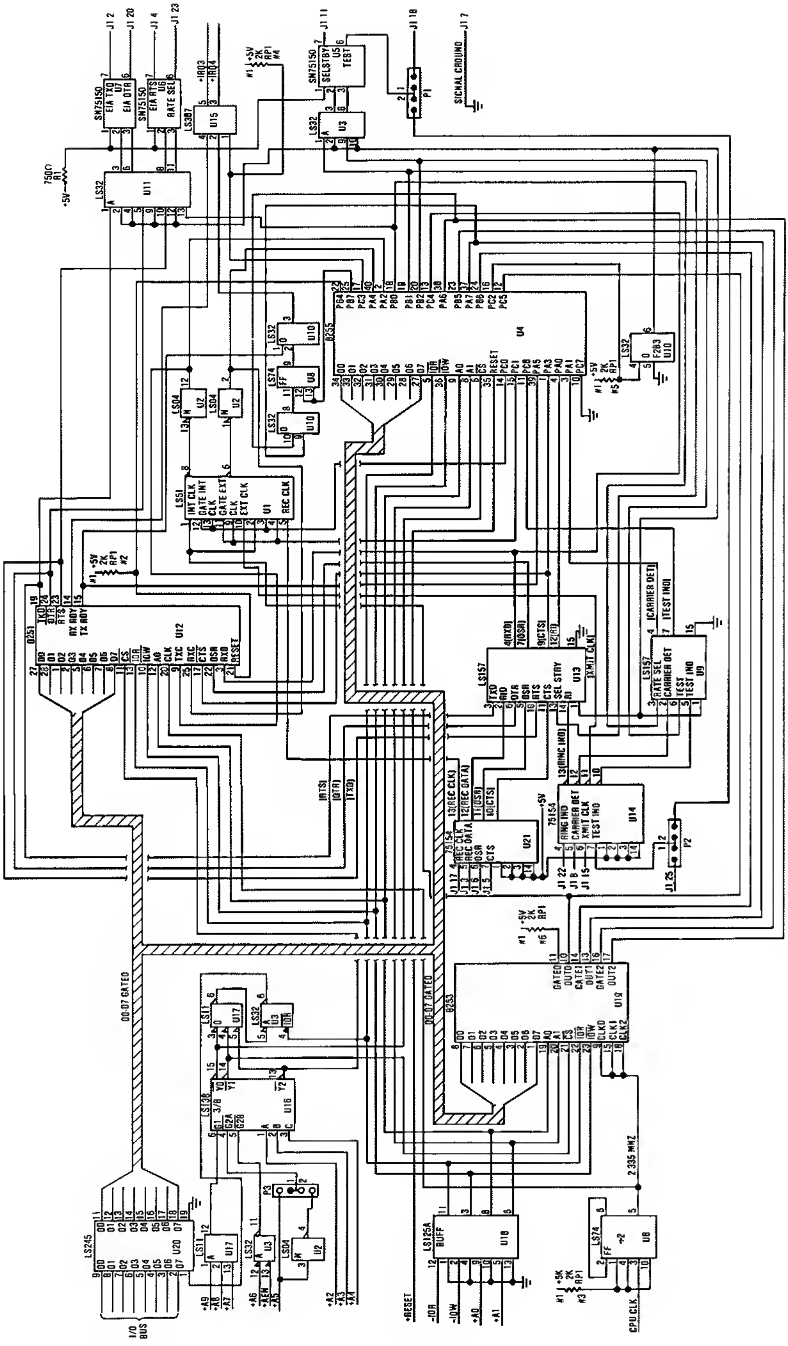


Asynchronous Communications Adapter (Sheet 1 of 1)

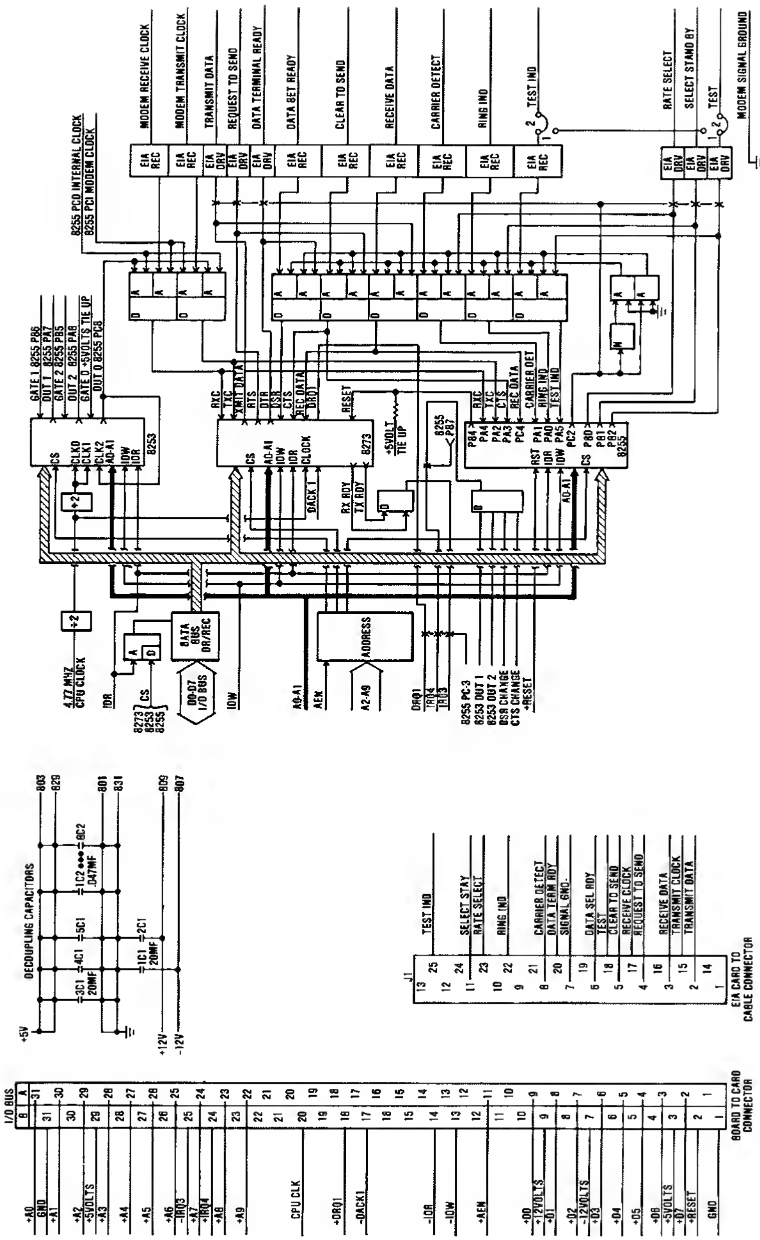


COMMUNICATION ADAPTER ADDRESS		
MODULE	HEX	NAME FUNCTION RESISTER
8255	3A0	PORT A REGISTER
	3A1	PORT B REGISTER
	3A2	PORT C REGISTER
	3A3	CONTROL REGISTER
8253	3A4	COUNTER 0 REGISTER
	3A5	COUNTER 1 REGISTER
	3A6	COUNTER 2 REGISTER
	3A7	TIMER MODE REGISTER
8251	3A8	DATA REGISTER
	3A9	CONTROL/STATUS REGIST

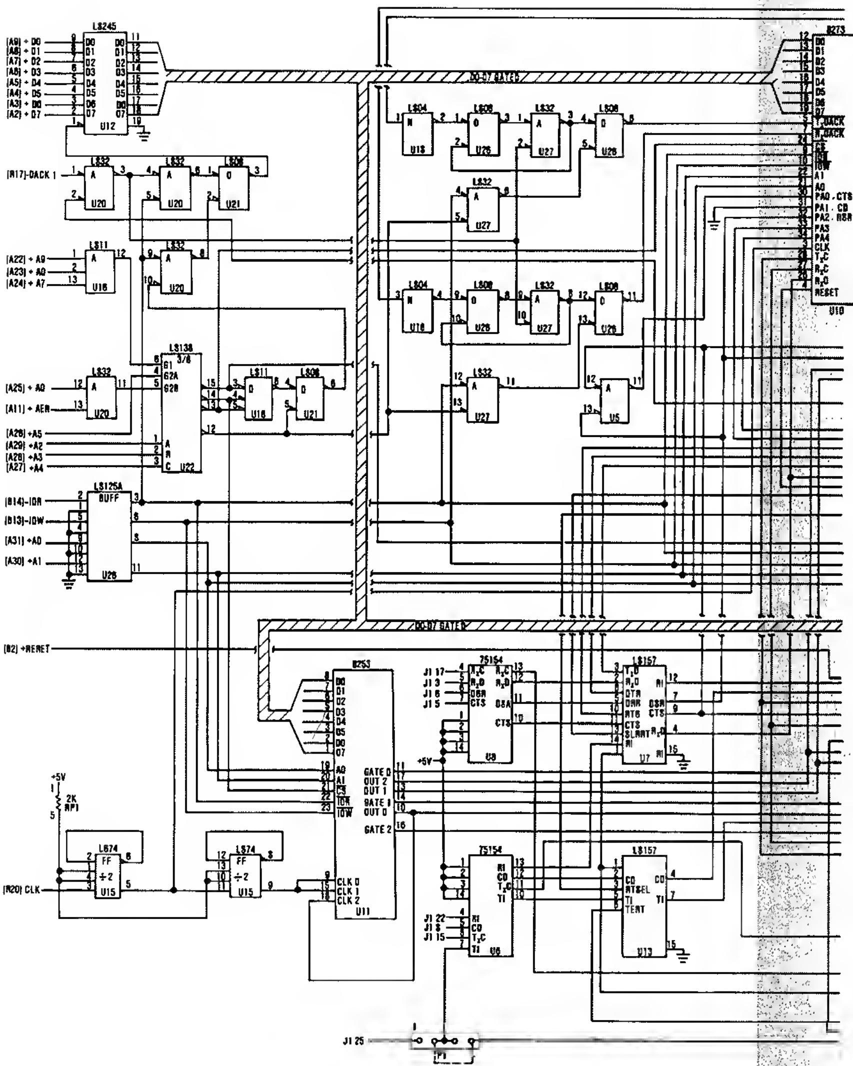
Binary Synchronous Communications Adapter (Sheet 1 of 2)



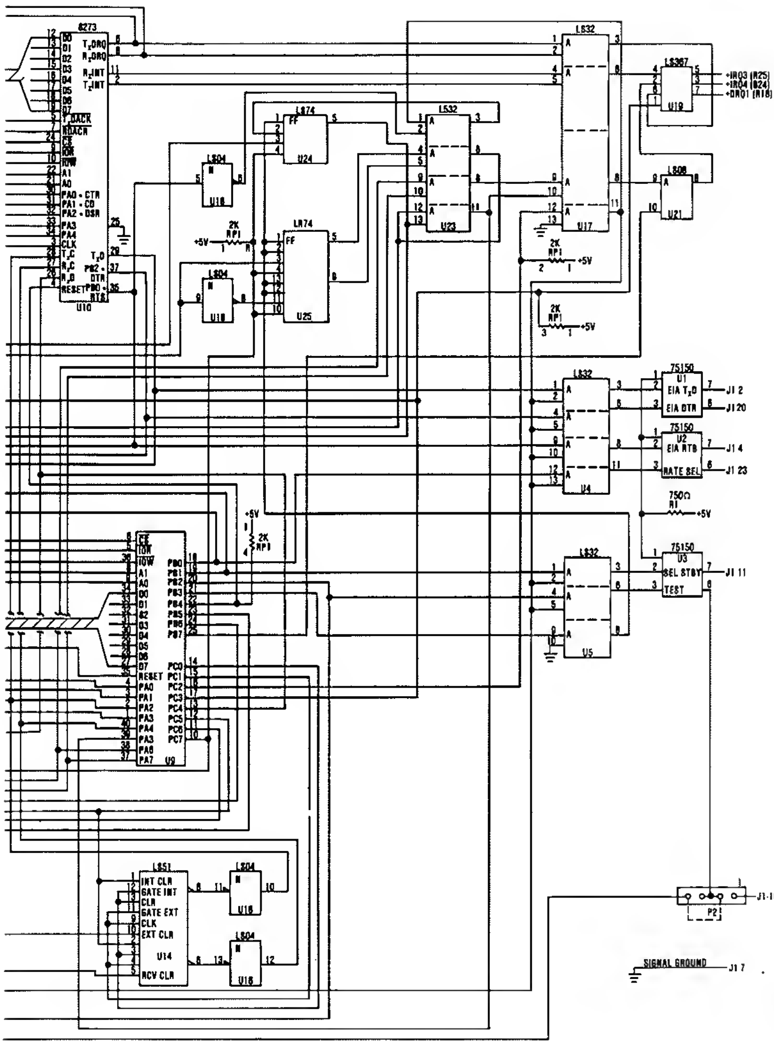
Binary Synchronous Communications Adapter (Sheet 2 of 2)



SDLC Communications Adapter (Sheet 1 of 2)



SDLC Communications Adapter (Sheet 1 of 2)



SDLC Communications Adapter (Sheet 2 of 2)

Notes:

APPENDIX E: SPECIFICATIONS

System Unit

Size:
- Length--19.6 in (500 mm)
- Depth--16.1 in (410 mm)
- Height--5.5 in (142 mm)

Weight:

20.9 lb (9.5 kg)	Without a diskette drive unit
25.0 lb (11.4 kg)	With one diskette drive unit

Power Cable:
- Length--6 ft (1.83 m)
- Size--18 AWG

Environment:
- Air Temperature
  - System ON, 60° to 90° F (15.6° to 32.2° C)
  - System OFF, 50° to 110° F (10° to 43° C)
- Humidity
  - System ON, 8% to 80%
  - System OFF, 20% to 80%

Heat Output:
- 1083 BTU/hr

Noise Level:
- 56 dB Without printer
- 66 dB With printer

Electrical:
- Nominal--120 Vac
- Minimum--104 Vac
- Maximum--127 Vac
- kVA--0.3175 (maximum)

Keyboard

Size:
- Length--19.6 in (500 mm)
- Depth--7.87 in (200 mm)
- Height--2.2 in (57 mm)

Weight:
- 6.5 lb (2.9 kg)

Color Display

Size:
- Length--15.4 in (392 mm)
- Depth--15.6 in (407 mm)
- Height--11.7 in (297 mm)

Weight:
- 26 lb (11.8 kg)

Heat Output:
- 240 BTU/hr

Power Cable:
- Length--6 ft (1.83 m)
- Size--18 AWG

Signal Cable:
- Length--5 ft (1.5 m)
- Size--22 AWG

Expansion Unit

Size:
- Length--19.6 in (500 mm)
- Depth--16.1 in (410 mm)
- Height--5.5 in (142 mm)

Weight:
- 33 lb (14.9 kg)

Power Cable:
- Length--6 ft (1.83 m)
- Size--18 AWG

Signal Cable:
- Length--3.28 ft (1 m)
- Size--22 AWG

Environment:
- Air Temperature
  - System ON, 60° to 90° F (15.6° to 32.2° C)
  - System OFF, 50° to 110° F (10° to 43° C)
- Humidity
  - System ON, 8% to 80%
  - System OFF, 20% to 80%

Heat Output:
- 717 BTU/hr

Electrical:
- Nominal--120 Vac
- Minimum--104 Vac
- Maximum--127 Vac

Monochrome Display

Size:
- Length--14.9 in (380 mm)
- Depth--13.7 in (350 mm)
- Height--11 in (280 mm)

Weight:
- 17.3 lb (7.9 kg)

Heat Output:
- 325 BTU/hr

Power Cable:
- Length--3 ft (0.914 m)
- Size--18 AWG

Signal Cable:
- Length--4 ft (1.22 m)
- Size--22 AWG

80 CPS Printers

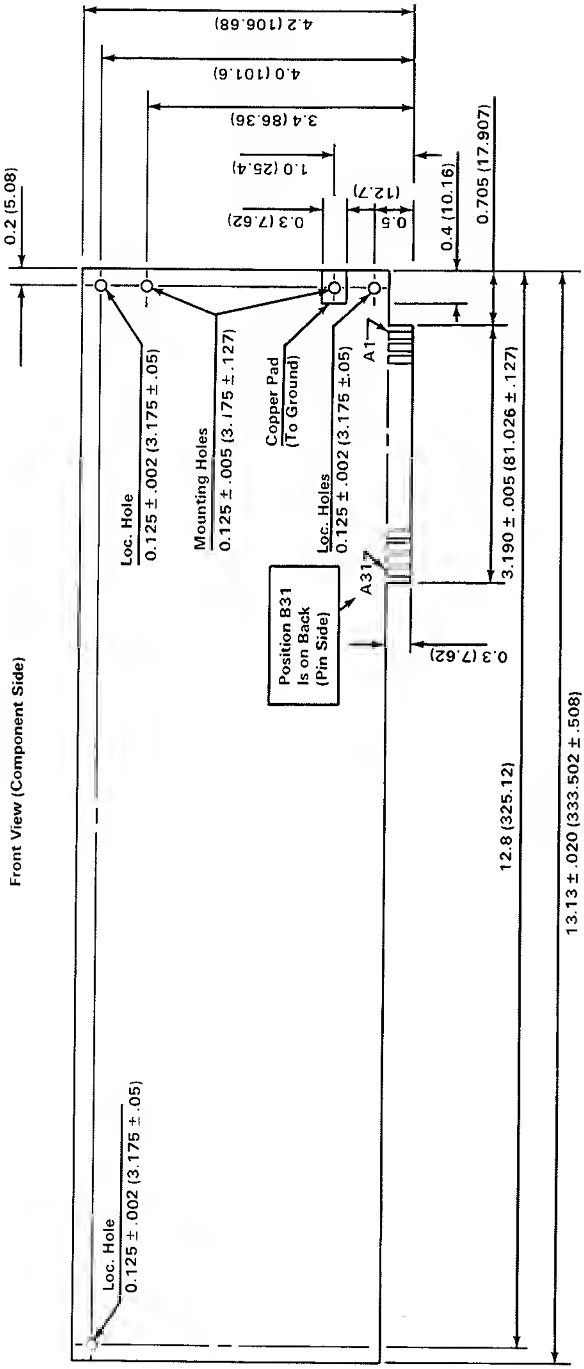
Size:
- Length--15.7 in (400 mm)
- Depth--14.5 in (370 mm)
- Height--4.3 in (110 mm)

Weight:
- 12.9 lb (5.9 kg)

Power Cable:
- Length--6 ft (1.83 mm)
- Size--22 AWG

Heat Output:
- 341 BTU/hr (maximum)

Electrical:
- Nominal--120 Vac
- Minimum--104 Vac
- Maximum--127 Vac



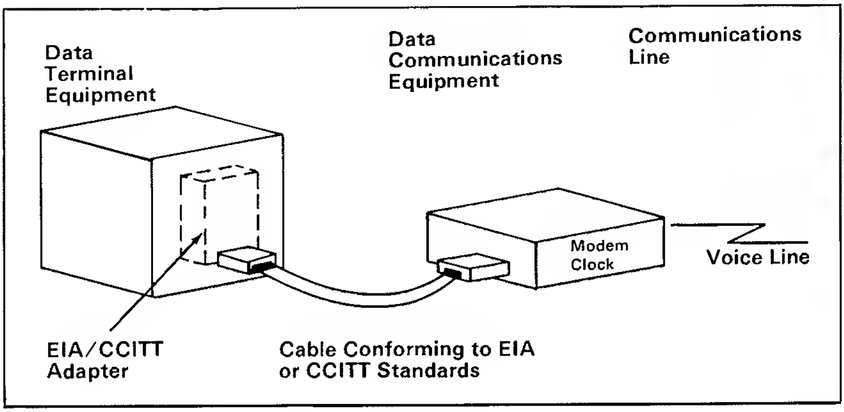
Notes:

1. All Card Dimensions are ±.010 (.254) Tolerance (With Exceptions Indicated on Drawing or in Notes).
2. Max. Card Length is 13.15 (334.01) Smaller Length is Permissible.
3. Loc. and Mounting Holes are Non-Plated Thru. (Loc. 3X, Mtg. 2X).
4. 31 Gold Tabs Each Side, 0.100 ± .0005 (2.54 ± .0127) Center to Center, 0.06 ± .0005 (1.524 ± .0127) Width.
5. Numbers in Parentheses are in Millimeters. All Others are in Inches.

APPENDIX F: COMMUNICATIONS

Information processing equipment used for communications is called data terminal equipment (DTE). Equipment used to connect the DTE to the communications line is called data communications equipment (DCE).

An adapter is used to connect the data terminal equipment to the data communications line as shown in the following illustration:



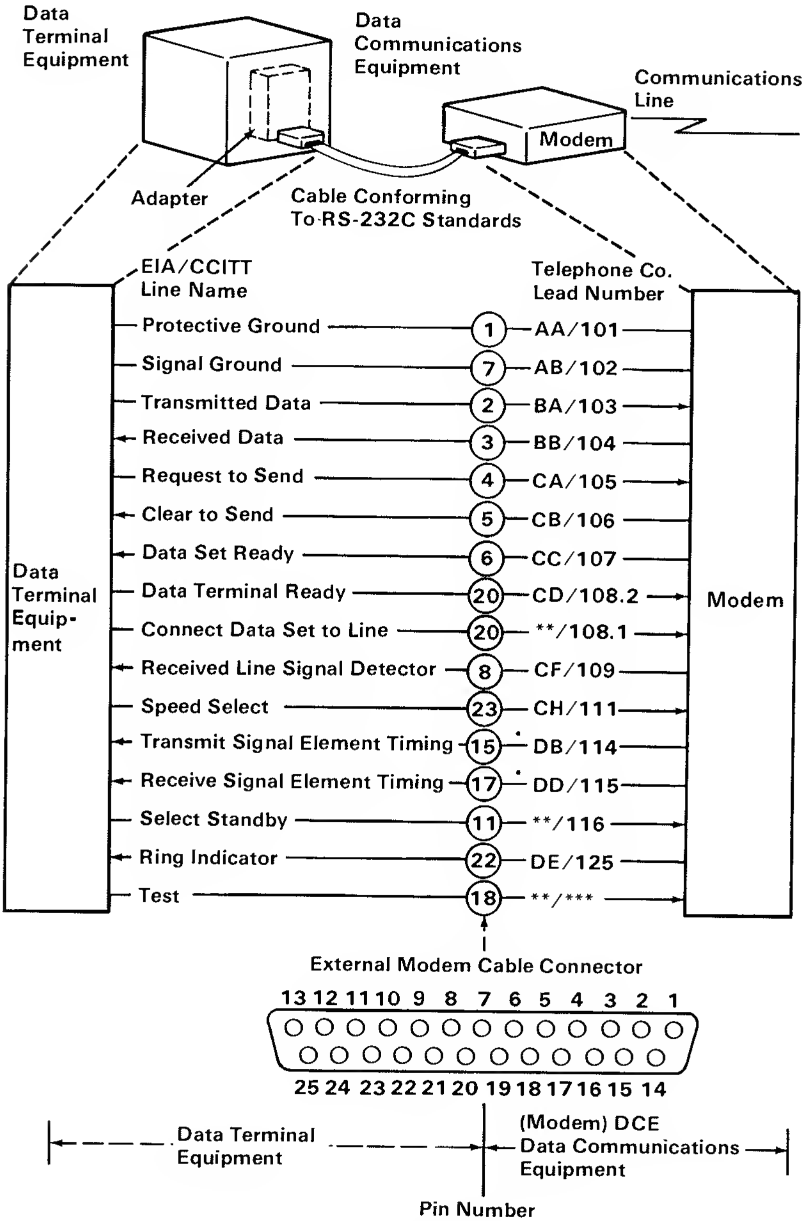
The EIA/CCITT adapter allows data terminal equipment to be connected to data communications equipment using EIA or CCITT standardized connections. An external modem is shown in this example; however, other types of data communications equipment can also be connected to data terminal equipment using EIA or CCITT standardized connections.

EIA standards are labeled RS-x (Recommended Standards-x) and CCITT standards are labeled V.x or X.x, where x is the number of the standard.

The EIA RS-232 interface standard defines the connector type, pin numbers, line names, and signal levels used to connect data terminal equipment to data communications equipment for the purpose of transmitting and receiving data. Since the RS-232 standard was developed, it has been revised three times. The three revised standards are the RS-232A, the RS-232B, and the presently used RS-232C.

The CCITT V.24 interface standard is equivalent to the RS-232C standard; therefore, the descriptions of the EIA standards also apply to the CCITT standards.

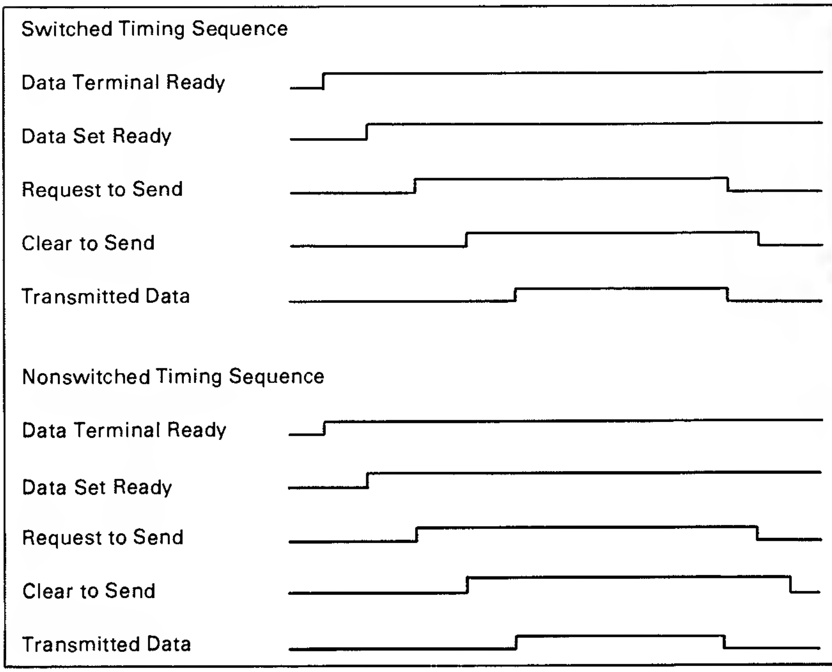
The following is an illustration of data terminal equipment connected to an external modem using connections defined by the RS-232C interface standard:



*Not used when business machine clocking is used.
**Not standardized by EIA (Electronics Industry Association).
***Not standardized by CCITT

Establishing a Communications Link

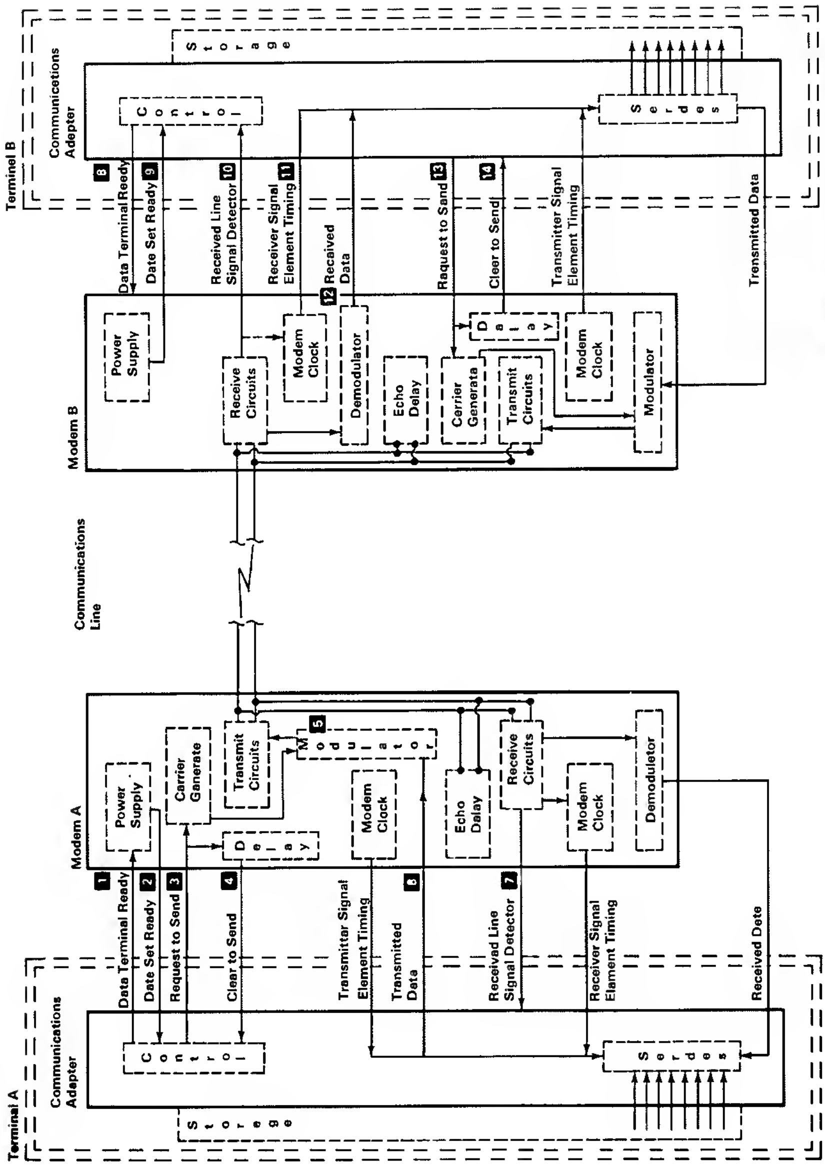
The following bar graphs represent normal timing sequences of operation during the establishment of communications for both switched (dial-up) and nonswitched (direct line) networks.



The following examples show how a link is established on a nonswitched point-to-point line, a nonswitched multipoint line, and a switched point-to-point line.

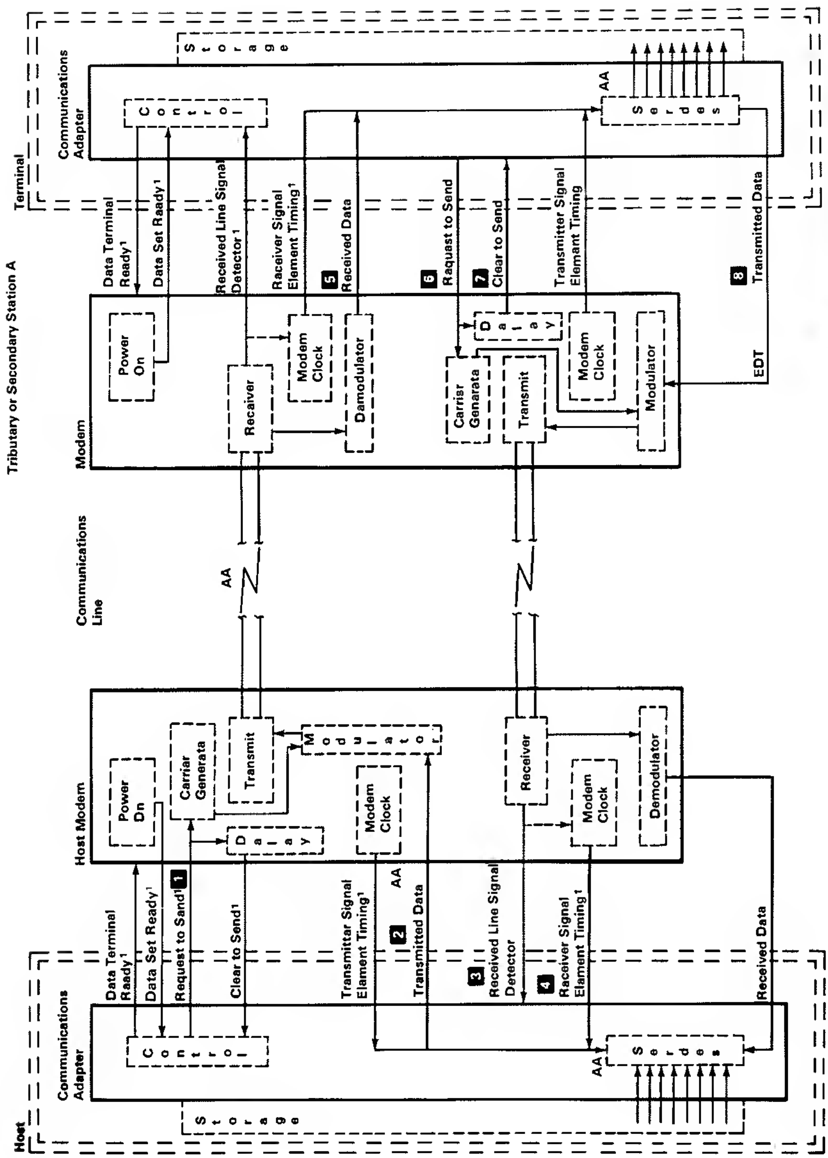
Establishing a Link on a Nonswitched Point-to-Point Line

1. The terminals at both locations activate the 'data terminal ready' lines 1 and 9.
2. Normally the 'data set ready' lines 2 and 9 from the modems are active whenever the modems are powered on.
3. Terminal A activates the 'request to send' line 3, which causes the modem at terminal A to generate a carrier signal.
4. Modem B detects the carrier, and activates the 'received line signal detector' line (sometimes called data carrier detect) 10. Modem B also activates the 'receiver signal element timing' line (sometimes called receive clock) 11 to send receive clock signals to the terminal. Some modems activate the clock signals whenever the modem is powered on.
5. After a specified delay, modem A activates the 'clear to send' line 4, which indicates to terminal A that the modem is ready to transmit data.
6. Terminal A serializes the data to be transmitted (through the serdes) and transmits the data one bit at a time (synchronized by the transmit clock) onto the 'transmitted data' line 6 to the modem.
7. The modem modulates the carrier signal with the data and transmits it to the modem B 5.
8. Modem B demodulates the data from the carrier signal and sends it to terminal B on the 'received data' line 12.
9. Terminal B deserializes the data (through the serdes) using the receive clock signals (on the 'receiver signal element timing' line) 11 from the modem.
10. After terminal A completes its transmission, it deactivates the 'request to send' line 3, which causes the modem to turn off the carrier and deactivate the 'clear to send' line 4.
11. Terminal A and modem A now become receivers and wait for a response from terminal B, indicating that all data has reached terminal B. Modem A begins an echo delay (50 to 150 milliseconds) to ensure that all echoes on the line have diminished before it begins receiving. An echo is a reflection of the transmitted signal. If the transmitting modem changed to receive too soon, it could receive a reflection (echo) of the signal it just transmitted.
12. Modem B deactivates the 'received line signal detector' line 10 and, if necessary, deactivates the receive clock signals on the 'receiver signal element timing, line 11.
13. Terminal B now becomes the transmitter to respond to the request from terminal A. To transmit data, terminal B activates the 'request to send' line 13, which causes modem B to transmit a carrier to modem A.
14. Modem B begins a delay that is longer than the echo delay at modem A before turning on the 'clear to send' line. The longer delay (called request-to-send to clear-to-send delay) ensures that modem A is ready to receive when terminal B begins transmitting data. After the delay, modem B activates the 'clear to send' line 14 to indicate that terminal B can begin transmitting its response.
15. After the echo delay at modem A, modem A senses the carrier from modem B (the carrier was activated in step 13 when terminal B activated the 'request to send' line) and activates the 'received line signal detector' line 7 to terminal A.
16. Modem A and terminal A are now ready to receive the response from terminal B. Remember, the response was not transmitted until after the request-to-send to clear-to-send delay at modem B (step 14).


Terminal B
Communications Adapter
Storage
Control
Serdes
8 Data Terminal Ready
9 Data Set Ready
10 Received Line Signal Detector
11 Receiver Signal Element Timing
12 Received Data
13 Request to Send
14 Clear to Send
Transmitter Signal Element Timing
Transmitted Data
Modem B
Power Supply
Receive Circuits
Modem Clock
Demodulator
Echo Delay
Carrier Generate
Transmit Circuits
Delay
Modulator
Communications Line
Modem A
Power Supply
Carrier Generate
Transmit Circuits
5 Modulator
Delay
Modem Clock
Echo Delay
Receive Circuits
Modem Clock
Demodulator
1 Data Terminal Ready
2 Data Set Ready
3 Request to Send
4 Clear to Send
Transmitter Signal Element Timing
6 Transmitted Data
7 Received Line Signal Detector
Receiver Signal Element Timing
Received Data
Terminal A
Communications Adapter
Control
Serdes
Storage

Establishing a Link on a Nonswitched Multipoint Line

1. The control station serializes the address for the tributary or secondary station (AA) and sends its address to the modem on the 'transmitted data' line 2.
2. Since the 'request to send' line and, therefore, the modem carrier, is active continuously 1, the modem immediately modulates the carrier with the address, and, thus, the address is transmitted to all modems on the line.
3. All tributary modems, including the modem for station A, demodulate the address and send it to their terminals on the 'received data' line 5.
4. Only station A responds to the address; the other stations ignore the address and continue monitoring their 'received data' line. To respond to the poll, station A activates its 'request to send' line 6, which causes the modem to begin transmitting a carrier signal.
5. The control station's modem receives the carrier and activates the 'received line signal detector, line 3 and the 'receiver signal element timing' line 4 (to send clock signals to the control station). Some modems activate the clock signals as soon as they are powered on.
6. After a short delay to allow the control station modem to receive the carrier, the tributary modem activates the 'clear to send' line 7.
7. When station A detects the active 'clear to send' line, it tansmits its response. (For this example, assume that station A has no data to send; therefore, it transmits an EOT 8.)
8. After transmitting the EOT, station A deactivates the 'request to send' line 6. This causes the modem to deactivate the carrier and the 'clear to send' line 7.
9. When the modem at the control station (host) detects the absence of the carrier, it deactivates the 'received line signal detector' line 3.
10. Tributary station A is now in receive mode waiting for the next poll or select transmission from the control station.

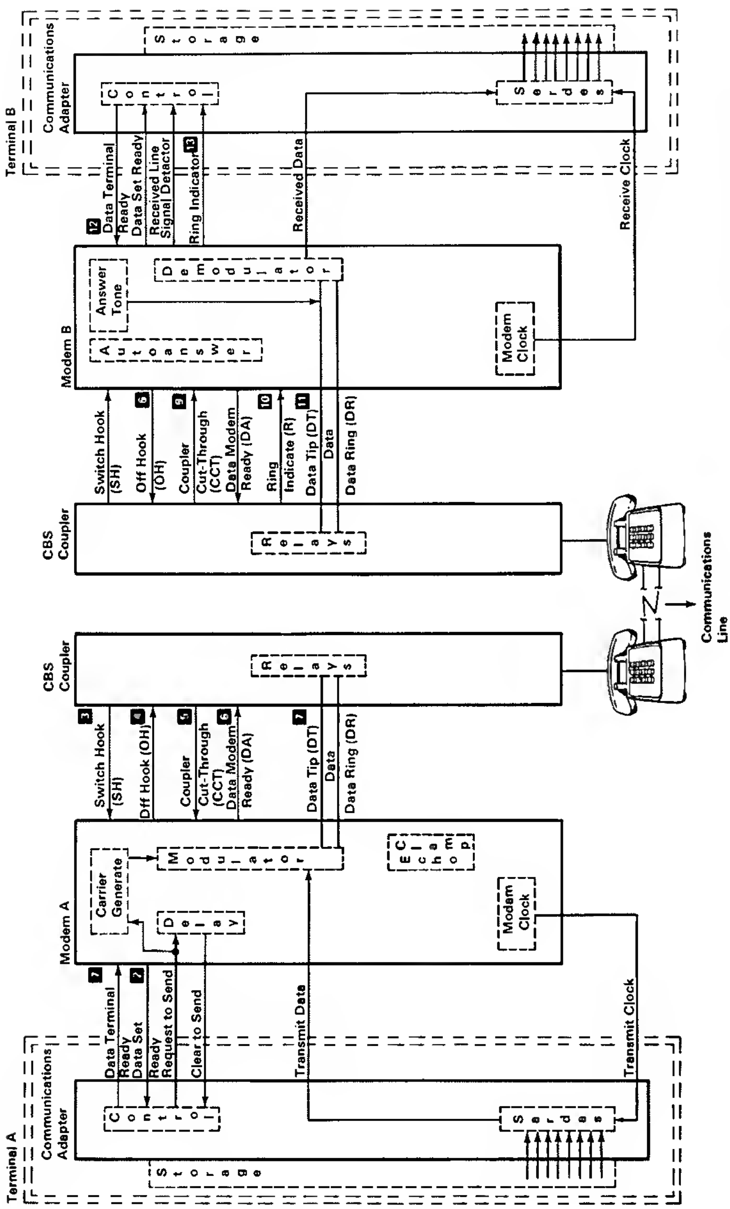


¹These lines are active continuously.

Establishing a Link on a Switched Point-To-Point Line

1. Terminal A is in communications mode; therefore, the 'data terminal ready' line 1 is active. Terminal B is in communication mode waiting for a call from terminal A.
2. When the terminal A operator lifts the telephone handset, the 'switch hook' line from the coupler is activated 3.
3. Modem A detects the 'switch hook' line and activates the 'off hook' line 4, which causes the coupler to connect the telephone set to the line and activate the 'coupler cut-through' line 5 to the modem.
4. Modem A activates the 'data modem ready' line 6 to the coupler (the 'data modem ready' line is on continuously in some modems).
5. The terminal A operator sets the exclusion key or talk/data switch to the talk position to connect the handset to the communications line. The operator then dials the terminal B number.
6. When the telephone at terminal B rings, the coupler activates the 'ring indicate' line to modem B 10. Modem B indicates that the 'ring indicate' line was activated by activating the 'ring indicator' line 13 to terminal B.
7. Terminal B activates the 'data terminal ready' line to modem B 12, which activates the autoanswer circuits in modem B. (The 'data terminal ready' line might already be active in some terminals.)
8. The autoanswer circuits in modem B activate the 'off hook' line to the coupler 8.
9. The coupler connects modem B to the communications line through the 'data tip' and 'data ring' lines 11 and activates the 'coupler cut-through' line 9 to the modem. Modem B then transmits an answer tone to terminal A.
10. The terminal A operator hears the tone and sets the exclusion key or talk/data switch to the data position (or performs an equivalent operation) to connect modem A to the communications line through the 'data tip' and 'data ring' lines 7.
11. The coupler at terminal A deactivates the 'switch hook' line 3. This causes modem A to activate the 'data set ready' line 2 indicating to terminal A that the modem is connected to the communications line.

The sequence of the remaining steps to establish the data link is the same as the sequence required on a nonswitched point-to-point line. When the terminals have completed their transmission, they both deactivate the 'data terminal ready' line to disconnect the modems from the line.



Terminal B
Communications Adapter
Storage
Control
Serdes
Data Terminal Ready
Data Set Ready
Received Line Signal Detector
Ring Indicator
Received Data
Receive Clock
Modem B
Answer Tone
Demodulator
Autoanswer
Modem Clock
Switch Hook (SH)
Off Hook (OH)
Coupler Cut-Through (CCT)
Data Modem Ready (DA)
Ring Indicate (R)
Data Tip (DT)
Data
Data Ring (DR)
CBS Coupler
Relays
Communications Line
CBS Coupler
Relays
Switch Hook (SH)
Off Hook (OH)
Coupler Cut-Through (CCT)
Data Modem Ready (DA)
Data Tip (DT)
Data
Data Ring (DR)
Modem A
Carrier Generate
Modulator
Echoplex
Delay
Modem Clock
Data Terminal Ready
Data Set Ready
Request to Send
Clear to Send
Transmit Data
Transmit Clock
Terminal A
Communications Adapter
Control
Serdes
Storage

Appendix F

Notes:

APPENDIX G: SWITCH SETTINGS

The following switch settings are divided between two groups. The first group contains the switch settings for the 16/64K system board. The second group contains the 64/256K system board switch settings.

Determine the system board type and refer to the appropriate group of switch settings for all applications.

Switch Settings (16KB-64KB CPU) G-3

Switch Settings (64KB-256KB CPU) G-29

Notes:

Switch Settings (16KB-64KB CPU)

System Board Switch Settings G-5
System Board Switch Settings G-5
5-1/4" Diskette Drives Switch Settings G-6
Display Type Switch Settings G-6
Math Coprocessor Switch Settings G-7

Memory Option Switch Settings G-8
16K Total Memory G-8
32K Total Memory G-8
48K Total Memory G-8
64K Total Memory G-8
96K Total Memory G-9
128K Total Memory G-10
160K Total Memory G-11
192K Total Memory G-12
224K Total Memory G-13
256K Total Memory G-14
288K Total Memory G-15
320K Total Memory G-16
352K Total Memory G-17
384K Total Memory G-18
416K Total Memory G-19
448K Total Memory G-20
480K Total Memory G-21
512K Total Memory G-22
544K Total Memory G-23
576K Total Memory G-24
608K Total Memory G-25
640K Total Memory G-26

Extender Card Switch Settings G-27

Appendix G

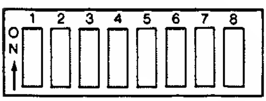
Notes:

Switch Setting Charts

System Board Switches

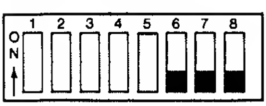
WARNING:	Before you change any switch settings, make a note of how the switches are presently set.

Switch Block 1



Switch	Function
1,7,8	Number of 5-1/4 inch diskette drives installed
2	Math Coprocessor
3,4	System board memory switches
5,6	Type(s) of display(s) connected

Switch Block 2



Switch	Function
1,2,3,4,5	Amount of memory options installed
6,7,8	Always in the Off position

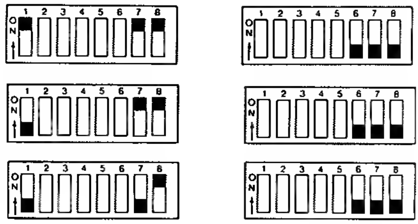
Number of 5-1/4 Inch Diskette Drives Installed

Switch Block 1 Switch Block 2

0 – Drives

1 – Drive

2 – Drives

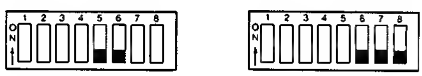


Type(s) of display(s) connected

WARNING: If an IBM Monochrome Display is connected to your system. Switch Block 1, switches 5 and 6, must always be Off. Damage to your display can result with any other switch settings.

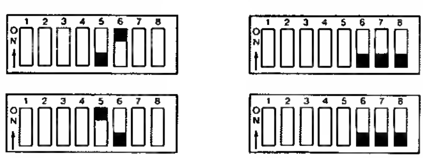
Switch Block 1 Switch Block 2

IBM Monochrome Display (or IBM Monochrome Display plus another display)



Switch Block 1 Switch Block 2

Color Display (Do not use if an IBM Monochrome Display is connected)

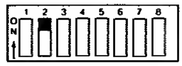
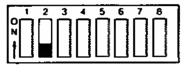
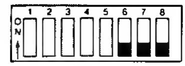


40x25 Mode

80x25 Mode

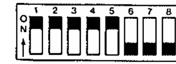
Note: The 40x25 mode means there will be 40 characters across the screen and 25 lines down the screen. The 80x25 mode means there will be 80 characters across the screen and 25 lines down the screen. The 80x25 mode, when used with home televisions and various displays, can cause loss of character quality.

Math Coprocessor

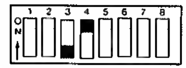
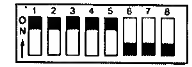
	Switch Block 1	Switch Block 2
With Math Coprocessor		
Without Math Coprocessor		

Memory Switch Settings
(16KB-64KB CPU) System Board

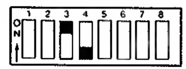
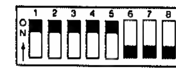
16K Total Memory

System Board Switches	Switch Block 1 	Switch Block 2 

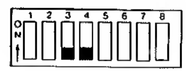
32K Total Memory

System Board Switches	Switch Block 1 	Switch Block 2 

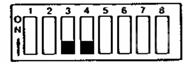
48K Total Memory

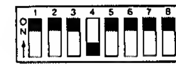
System Board Switches	Switch Block 1 	Switch Block 2 

64K Total Memory

System Board Switches	Switch Block 1 	Switch Block 2 

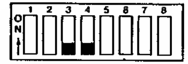
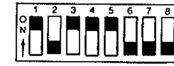
96K Total Memory
32K + (64K on System Board)

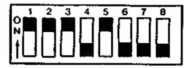
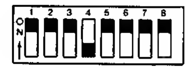
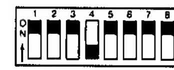
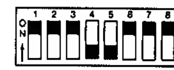
System Board Switches	Switch Block 1 	Switch Block 2 

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 32K option			

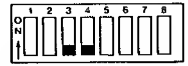
Appendix G

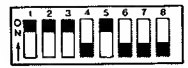
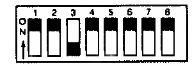
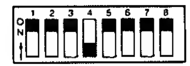
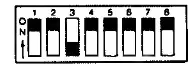
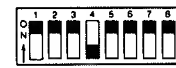
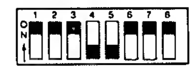
128K Total Memory
64K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

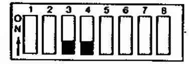
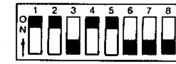
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed			
1 - 64K option			
2 - 32K options			 

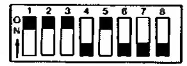
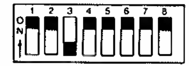
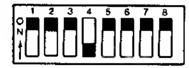
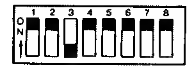
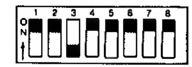
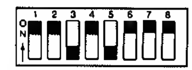
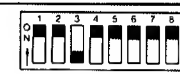
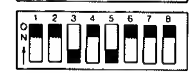
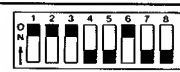
160K Total Memory
96K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

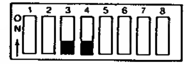
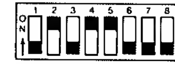
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed 1 - 32K option			
1 - 64K option 1 - 32K option			
3 - 32K options			  

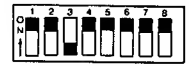
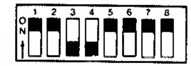
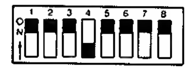
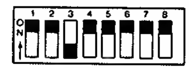
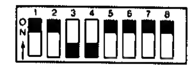
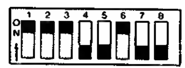
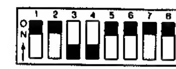
192K Total Memory
128K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K option installed 1 - 64K option			
2 - 64K options		 	
1 - 64/256K option with 64K installed 2 - 32K options			 
1 - 64K option 2 - 32K options			 
1 - 64/256K option with 128K installed			

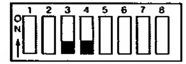
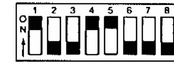
224K Total Memory
160K + (64K on System Board)

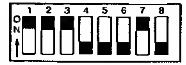
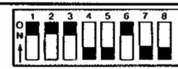
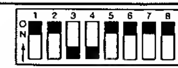
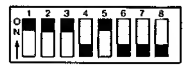
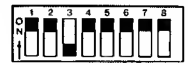
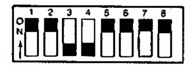
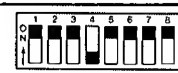
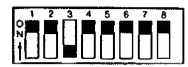
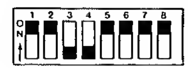
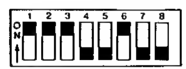
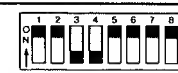
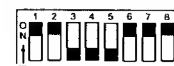
System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed 1 - 64K option 1 - 32K option			
2 - 64K options 1 - 32K option		 	
1 - 64/256K option with 128K installed 1 - 32K option			

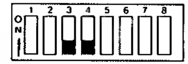
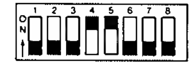
Appendix G

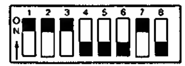
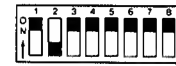
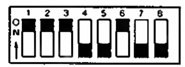
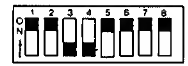
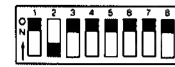
256K Total Memory
192K + (64K on System Board)

System Board Switches	Switch Block 1		Switch Block 2	
				

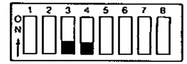
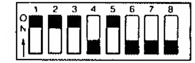
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed			
1 - 64/256K option with 128K installed 1 - 64K option			
1 - 64/256K option with 64K installed 2 - 64K options		 	
3 - 64K options		  	
1 - 64/256K option with 128K installed 2 - 32K options			 

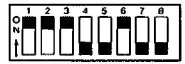
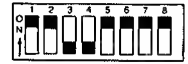
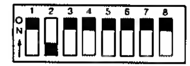
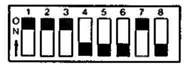
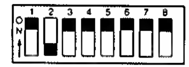
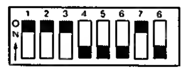
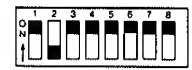
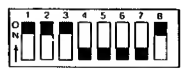
288K Total Memory
224K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

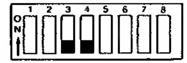
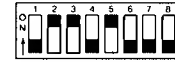
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 1 - 32K option			
1 - 64/256K option with 128K installed 1 - 64K option 1 - 32K option			

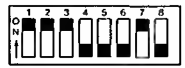
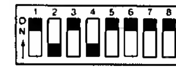
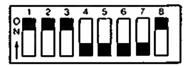
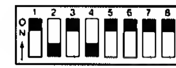
320K Total Memory
256K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 128K installed 2 - 64K options		 	
1 - 64/256K option with 192K installed 1 - 64K option			
1 - 64/256K option with 192K installed 2 - 32K options			 
1 - 64/256K option with 256K installed			

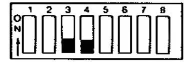
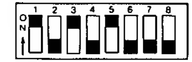
352K Total Memory
288K + (64K on System Board)

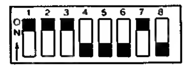
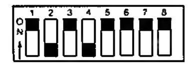
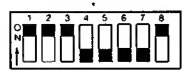
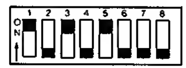
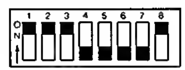
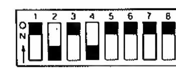
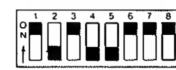
System Board Switches	Switch Block 1		Switch Block 2	

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 1 - 64K option 1 - 32K option			
1 - 64/256K option with 256K installed 1 - 32K option			

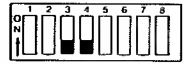
Appendix G

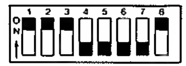
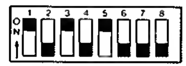
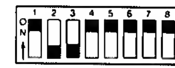
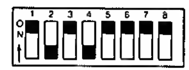
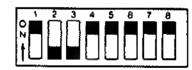
384K Total Memory
320K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 2 - 64K options		 	
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed	 		
1 - 64/256K option with 256K installed 1 - 64K option			
1 - 64/256K option with 256K installed 2 - 32K options			 

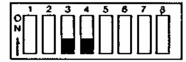
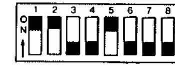
416K Total Memory
352K + (64K on System Board)

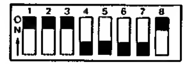
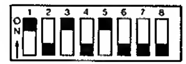
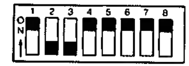
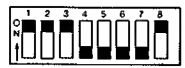
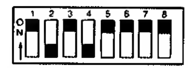
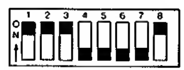
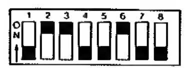
System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 32K option	 		
1 - 64/256K option with 256K installed 1 - 64K option 1 - 32K option			

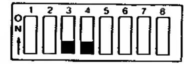
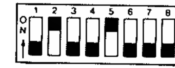
Appendix G

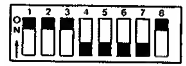
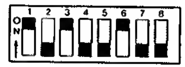
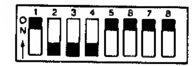
448K Total Memory
384K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
	Switch Block 1 	Switch Block 2 

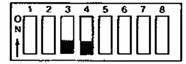
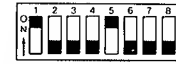
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 64K option	 		
1 - 64/256K option with 256K installed 2 - 64K options		 	
1 - 64/256K option with 256K installed 1 - 64/256K option with 128K installed	 		

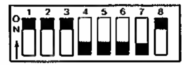
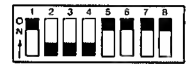
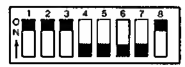
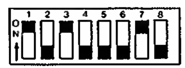
480K Total Memory
416K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

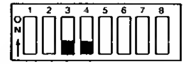
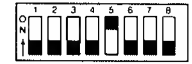
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 128K installed 1 - 32K option	 		

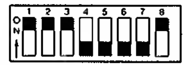
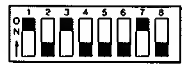
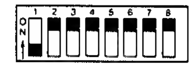
512K Total Memory
448K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	64K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 128K installed 1 - 64K option	 		
1 - 64/256K option with 256K installed 1 - 64/256K option with 192K installed	 		

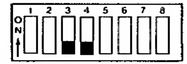
544K Total Memory
480K + (64K on System Board)

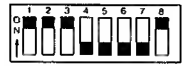
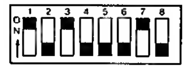
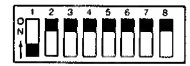
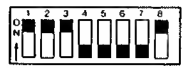
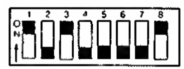
System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 192K installed 1 - 32K option	 		

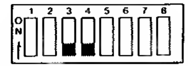
Appendix G

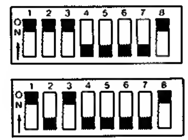
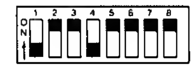
576K Total Memory
512K + (64K on System Board)

System Board Switches	Switch Block 1 	Switch Block 2 

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 192K installed 1 - 64K option	 		
2 - 64/256K option with 256K installed	 		

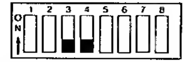
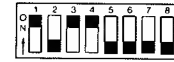
608K Total Memory
544K + (64K on System Board)

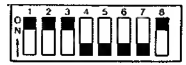
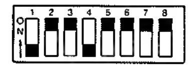
System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
2 - 64/256K option with 256K installed 1 - 32K option			

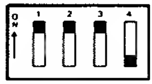
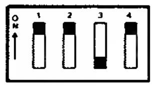
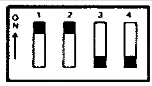
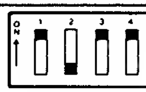
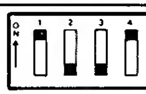
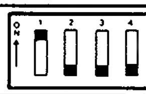
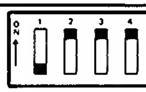
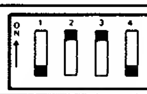
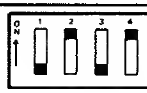
Appendix G

640K Total Memory
576K + (64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
2 - 64/256K option with 256K installed 1 - 64K option	 		
2 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed	  		

Extender Card Switch Settings

System Memory	Extender Card Switch Block	Memory Segment
16K to 64K		1
96K to 128K		2
160K to 192K		3
224K to 256K		4
288K to 320K		5
352K to 384K		6
416K to 448K		7
480K to 512K		8
544K to 576K		9
608K to 640K		A

Notes:

Switch Settings (64KB-256KB CPU)

System Board Switch Settings G-31
System Board Switch Settings G-31
5-1/4" Diskette Drives Switch Settings G-32
Display Type Switch Settings G-32
Math Coprocessor Switch Settings G-32

Memory Option Switch Settings G-34
64K Total Memory G-34
128K Total Memory G-34
192K Total Memory G-34
256K Total Memory G-34
288K Total Memory G-35
320K Total Memory G-36
352K Total Memory G-37
384K Total Memory G-38
416K Total Memory G-39
448K Total Memory G-40
480K Total Memory G-41
512K Total Memory G-42
544K Total Memory G-43
576K Total Memory G-44
608K Total Memory G-45
640K Total Memory G-46

Extender Card Switch Settings G-47

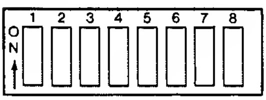
Notes:

Switch Setting Charts

System Board Switches

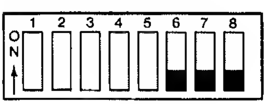
WARNING: Before you change any switch settings, make a note of how the switches are presently set.

Switch Block 1



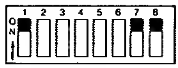
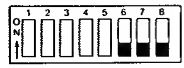
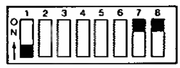
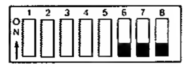
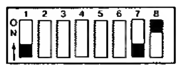
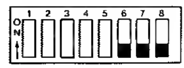
Switch	Function
1,7,8	Number of 5-1/4 inch diskette drives installed
2	Math Coprocessor
3,4	System board memory switches
5,6	Type(s) of display(s) connected

Switch Block 2



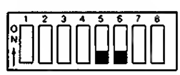
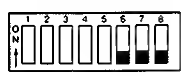
Switch	Function
1,2,3,4,5	Amount of memory options installed
6,7,8	Always in the Off position

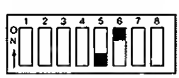
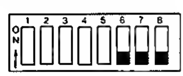
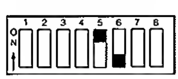
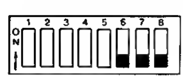
Number of 5-1/4 Inch Diskette Drives Installed

	Switch Block 1	Switch Block 2
0 – Drives		
1 – Drive		
2 – Drives		

Type(s) of display(s) connected

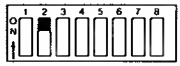
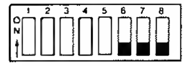
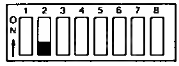
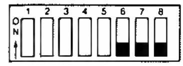
WARNING: If an IBM Monochrome Display is connected to your system. Switch Block 1, switches 5 and 6, must always be Off. Damage to your display can result with any other switch settings.

	Switch Block 1	Switch Block 2
IBM Monochrome Display (or IBM Monochrome Display plus another display)		

	Switch Block 1	Switch Block 2	
Color Display (Do not use if an IBM Monochrome Display is connected)			40x25 Mode
			80x25 Mode

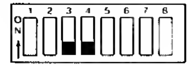
Note: The 40x25 mode means there will be 40 characters across the screen and 25 lines down the screen. The 80x25 mode means there will be 80 characters across the screen and 25 lines down the screen. The 80x25 mode, when used with home televisions and various displays, can cause loss of character quality.

Math Coprocessor

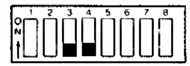
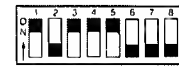
	Switch Block 1	Switch Block 2
With Math Coprocessor		
Without Math Coprocessor		

Memory Switch Settings
(64KB-256KB CPU) System Board

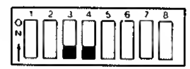
64K Total Memory

System Board Switches	Switch Block 1	Switch Block 2
		

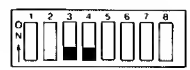
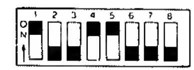
128K Total Memory

System Board Switches	Switch Block 1	Switch Block 2
		

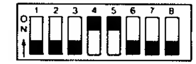
192K Total Memory

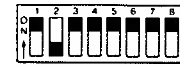
System Board Switches	Switch Block 1	Switch Block 2
		

256K Total Memory

System Board Switches	Switch Block 1	Switch Block 2
		

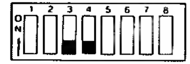
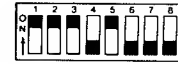
288K Total Memory
32K + (256K on System Board)

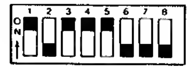
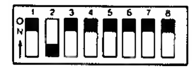
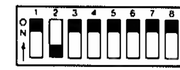
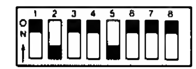
System Board Switches	Switch Block 1		Switch Block 2	
				

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 32K option			

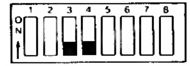
Appendix G

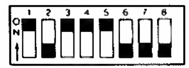
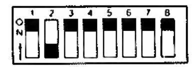
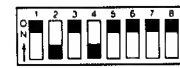
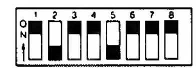
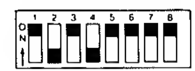
320K Total Memory
64K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed			
1 - 64K option			
2 - 32K options			 

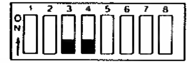
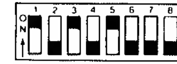
352K Total Memory
96K + (256K on System Board)

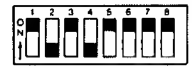
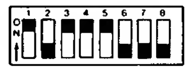
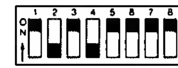
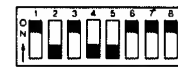
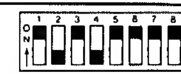
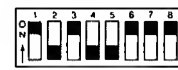
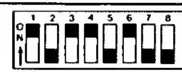
System Board Switches	Switch Block 1 	Switch Block 2 

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed 1 - 32K option			
1 - 64K option 1 - 32K option			
3 - 32K options			  

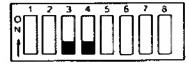
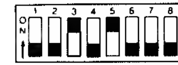
Appendix G

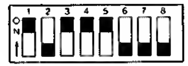
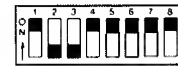
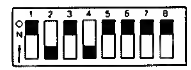
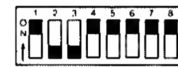
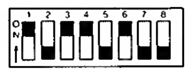
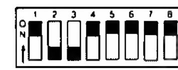
384K Total Memory
128K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K option installed 1 - 64K option			
2 - 64K options		 	
1 - 64/256K option with 64K installed 2 - 32K options			 
1 - 64K option 2 - 32K options			 
1 - 64/256K option with 128K installed			

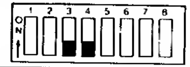
416K Total Memory
160K + (256K on System Board)

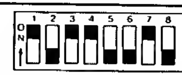
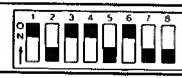
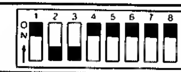
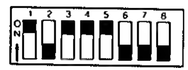
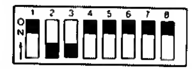
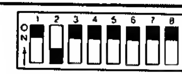
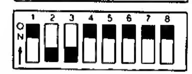
System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed 1 - 64K option 1 - 32K option			
2 - 64K options 1 - 32K option		 	
1 - 64/256K option with 128K installed 1 - 32K option			

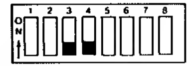
Appendix G

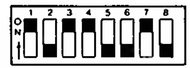
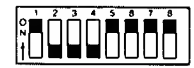
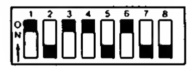
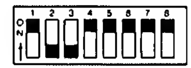
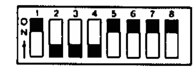
448K Total Memory
192K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

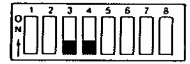
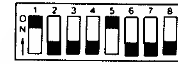
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed			
1 - 64/256K option with 128K installed 1 - 64K option			
1 - 64/256K option with 64K installed 2 - 64K options		 	
3 - 64K options		  	
1 - 64/256K option with 128 installed 2 - 32K options			 

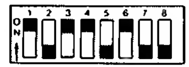
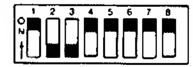
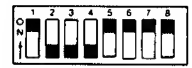
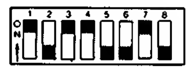
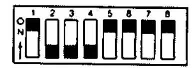
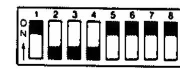
480K Total Memory
224K + (256K on System Board)

System Board Switches	Switch Block 1 	Switch Block 2 

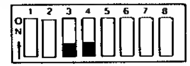
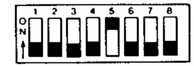
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 1 - 32K option			
1 - 64/256K option with 128K installed 1 - 64K option 1 - 32K option			

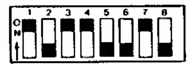
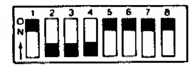
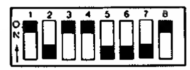
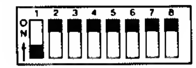
512K Total Memory
256K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

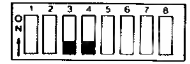
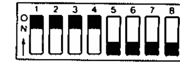
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 128K installed 2 - 64K options		 	
1 - 64/256K option with 192K installed 1 - 64K option			
1 - 64/256K option with 192K installed 2 - 32K options			 
1 - 64/256K option with 256K installed			

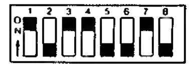
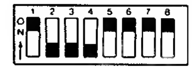
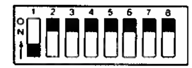
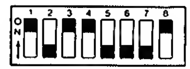
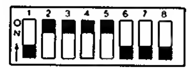
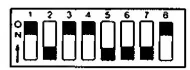
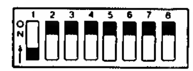
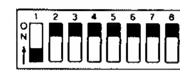
544K Total Memory
288K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

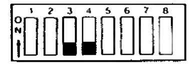
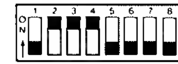
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 1 - 64K option 1 - 32K option			
1 - 64/256K option with 256K installed 1 - 32K option			

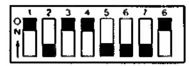
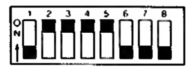
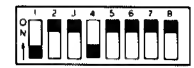
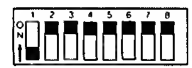
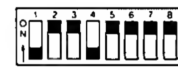
576K Total Memory
320K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

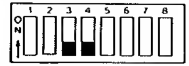
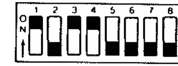
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 2 - 64K options		 	
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed	 		
1 - 64/256K option with 256K installed 1 - 64K option			
1 - 64/256K option with 256K installed 2 - 32K options			 

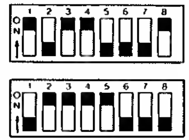
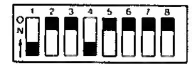
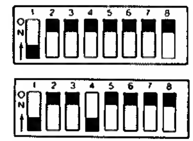
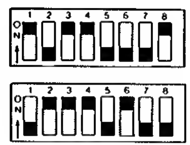
608K Total Memory
352K + (256K on System Board)

System Board Switches	Switch Block 1		Switch Block 2	
				

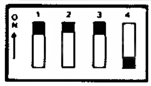
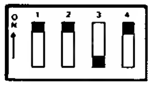
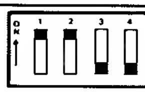
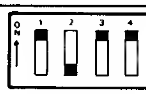
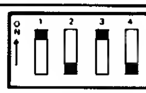
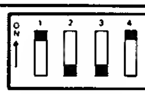
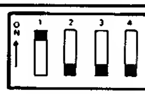
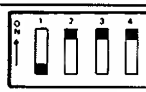
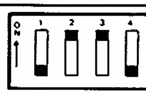
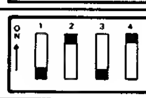
	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 32K option	 		
1 - 64/256K option with 256K installed 1 - 64K option 1 - 32K option			

640K Total Memory
384K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 64K option			
1 - 64/256K option with 256K installed 2 - 64K options			
1 - 64/256K option with 256K installed 1 - 64/256K option with 128K installed			

Extender Card Switch Settings

System Memory	Extender Card Switch Block	Memory Segment
16K to 64K		1
96K to 128K		2
160K to 192K		3
224K to 256K		4
288K to 320K		5
352K to 384K		6
416K to 448K		7
480K to 512K		8
544K to 576K		9
608K to 640K		A

Notes:

GLOSSARY

μs: Microsecond.

adapter: An auxiliary system or unit used to extend the operation of another system.

address bus: One or more conductors used to carry the binary-coded address from the microprocessor throughout the rest of the system.

all points addressable (APA): A mode in which all points on a displayable image can be controlled by the user.

alpanumeric (A/N): Pertaining to a character set that contains letters, digits, and usually other characters, such as punctuation marks. Synonymous with alphanumeric.

American Standard Code for Information Interchange (ASCII): The standard code, using a coded character set consisting of 7-bit coded characters (8 bits including parity check), used for information interchange among data processing systems, data communication systems and associated equipment. The ASCII set consists of control characters and graphic characters.

A/N: Alphanumeric.

analog: (1) pertaining to data in the form of continuously variable physical quantities. (2) Contrast with digital.

AND: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,...,then the AND of P, Q, R,...is true if all statements are true, false if any statement is false.

APA: All points addressable.

ASCII: American Standard Code for Information Interchange.

assembler: A computer program used to assemble. Synonymous with assembly program.

asynchronous communications: A communication mode in which each single byte of data is synchronized, usually by the addition of start/stop bits.

BASIC: Beginner's all-purpose symbolic instruction code.

basic input/output system (BIOS): Provides the device level control of the major I/O devices in a computer system, which provides an operational interface to the system and relieves the programmer from concern over hardware device characteristics.

baud: (1) A unit of signaling speed equal to the number of discrete conditions or signal events per second. For example, one baud equals one-half dot cycle per second in Morse code, one bit per second in a train of binary signals, and one 3-bit value per second in a train of signals each of which can assume one of eight different states. (2) In asynchronous transmission, the unit of modulation rate corresponding to one unit of interval per second; that is, if the duration of the unit interval is 20 milliseconds, the modulation rate is 50 baud.

BCC: Block-check character.

beginner's all-purpose symbolic instruction code (BASIC): A programming language with a small repertoire of commands and a simple syntax, primarily designed for numerical application.

binary: (1) Pertaining to a selection, choice, or condition that has two possible values or states. (2) Pertaining to a fixed radix numeration system having a radix of two.

binary digit: (1) In binary notation, either of the characters 0 or 1. (2) Synonymous with bit.

binary notation: Any notation that uses two different characters, usually the binary digits 0 and 1.

binary synchronous communications (BSC): A standardized procedure, using a set of control characters and control character sequences for synchronous transmission of binary-coded data between stations.

BIOS: Basic input/output system.

bit: In binary notation, either of the characters 0 or 1.

bits per second (bps): A unit of measurement representing the number of discrete binary digits which can be transmitted by a device in one second.

block-check character (BCC): In cyclic redundancy checking, a character that is transmitted by the sender after each message block and is compared with a block-check character computed by the receiver to determine if the transmission was successful.

boolean operation: (1) Any operation in which each of the operands and the result take one of two values. (2) An operation that follows the rules of boolean algebra.

bootstrap: A technique or device designed to bring itself into a desired state by means of its own action; that is, a machine routine whose first few instructions are sufficient to bring the rest of itself into the computer from an input device.

bps: Bits per second.

BSC: Binary synchronous communications.

buffer: (1) An area of storage that is temporarily reserved for use in performing an input/output operation, into which data is read or from which data is written. Synonymous with I/O area. (2) A portion of storage for temporarily holding input or output data.

bus: One or more conductors used for transmitting signals or power.

byte: (1) A binary character operated upon as a unit and usually shorter than a computer word. (2) The representation of a character.

CAS: Column address strobe.

cathode ray tube (CRT): A vacuum tube display in which a beam of electrons can be controlled to form alphanumeric characters or symbols on a luminescent screen, for example by use of a dot matrix.

cathode ray tube display (CRT display): (1) A device that presents data in visual form by means of controlled electron beams. (2) The data display produced by the device as in (1).

CCITT: Comite Consultatif International Telegrafique et Telephonique.

central processing unit (CPU): A functional unit that consists of one or more processors and all or part of internal storage.

channel: A path along which signals can be sent; for example, data channel or I/O channel.

characters per second (cps): A standard unit of measurement for printer output.

code: (1) A set of unambiguous rules specifying the manner in which data may be represented in a discrete form. Synonymous with coding scheme. (2) A set of items, such as abbreviations, representing the members of another set. (3) Loosely, one or more computer programs, or part of a computer program. (4) To represent data or a computer program in a symbolic form that can be accepted by a data processor.

column address strobe (CAS): A signal that latches the column addresses in a memory chip.

Comite Consultatif International Telegrafique et Telephonique (CCITT): Consultative Committee on International Telegraphy and Telephony.

computer: A functional unit that can perform substantial computation, including numerous arithmetic operations, or logic operations, without intervention by a human operator during the run.

configuration: (1) The arrangement of a computer system or network as defined by the nature, number, and the chief characteristics of its functional units. More specifically, the term configuration may refer to a hardware configuration or a software configuration. (2) The devices and programs that make up a system, subsystem, or network.

conjunction: (1) The boolean operation whose result has the boolean value 1 if, and only if, each operand has the boolean value 1. (2) Synonymous with AND operation.

contiguous: (1) Touching or joining at the edge or boundary. (2) Adjacent.

CPS: Characters per second.

CPU: Central processing unit.

CRC: Cyclic redundancy check.

CRT: Cathode ray tube.

CRT display: Cathode ray tube display.

CTS: Clear to send. Associated with modem control.

cyclic redundancy check (CRC): (1) A redundancy check in which the check key is generated by a cyclic algorithm. (2) A system of error checking performed at both the sending and receiving station after a block-check character has been accumulated.

cylinder: (1) The set of all tracks with the same nominal distance from the axis about which the disk rotates. (2) The tracks of a disk storage device that can be accessed without repositioning the access mechanism.

daisy-chained cable: A type of cable that has two or more connectors attached in series.

data: (1) A representation of facts, concepts, or instructions in a formalized manner suitable for communication, interpretation, or processing by humans or automatic means. (2) Any representations, such as characters or analog quantities, to which meaning is, or might be assigned.

decoupling capacitor: A capacitor that provides a low-impedance path to ground to prevent common coupling between states of a circuit.

Deutsche Industrie Norm (DIN): (1) German Industrial Norm. (2) The committee that sets German dimension standards.

digit: (1) A graphic character that represents an integer, for example, one of the characters 0 to 9. (2) A symbol that represents one of the non-negative integers smaller than the radix. For example, in decimal notation, a digit is one of the characters from 0 to 9.

digital: (1) Pertaining to data in the form of digits. (2) Contrast with analog.

DIN: Deutsche Industrie Norm.

DIN connector: One of the connectors specified by the DIN standardization committee.

DIP: Dual in-line package.

direct memory access (DMA): A method of transferring data between main storage and I/O devices that does not require processor intervention.

disk: Loosely, a magnetic disk unit.

diskette: A thin, flexible magnetic disk and a semi-rigid protective jacket, in which the disk is permanently enclosed. Synonymous with flexible disk.

DMA: Direct memory access.

DSR: Data set ready. Associated with modem control.

DTR: Data terminal ready. Associated with modem control.

dual in-line package (DIP): A widely used container for an integrated circuit. DIPs are pins usually in two parallel rows. These pins are spaced 1/10 inch apart and come in different configurations ranging from 14-pin to 40-pin configurations.

EBCDIC: Extended binary-coded decimal interchange code.

ECC: Error checking and correction.

edge connector: A terminal block with a number of contacts attached to the edge of a printed circuit board to facilitate plugging into a foundation circuit.

EIA: Electronic Industries Association.

EIA/CCITT: Electronics Industries Association/Consultative Committee on International Telegraphy and Telephony.

end-of-text-character (ETX): A transmission control character used to terminate text.

end-of-transmission character (EOT): A transmission control character used to indicate the conclusion of a transmission, which may have included one or more texts and any associated message headings.

EOT: End-of-transmission character.

EPROM: Erasable programmable read-only memory.

erasable programmable read-only memory (EPROM): A storage device whose contents can be changed by electrical means. EPROM information is not destroyed when power is removed.

error checking and correction (ECC): The detection and correction of all single-bit, double-bit, and some multiple-bit errors.

ETX: End-of-text character.

extended binary-coded decimal interchange code (EBCDIC): A set of 256 characters, each represented by eight bits.

flexible disk: Synonym for diskette.

firmware: Memory chips with integrated programs already incorporated on the chip.

gate: (1) A device or circuit that has no output until it is triggered into operation by one or more enabling signals, or until an input signal exceeds a predetermined threshold amplitude. (2) A signal that triggers the passage of other signals through a circuit.

graphic: A symbol produced by a process such as handwriting, drawing, or printing.

hertz (Hz): A unit of frequency equal to one cycle per second.

hex: Abbreviation for hexadecimal.

hexadecimal: Pertaining to a selection, choice, or condition that has 16 possible values or states. These values or states usually contain 10 digits and 6 letters, A through F. Hexadecimal digits are equivalent to a power of 16.

high-order position: The leftmost position in a string of characters.

Hz: Hertz.

interface: A device that alters or converts actual electrical signals between distinct devices, programs, or systems.

k: An abbreviation for the prefix kilo; that is, 1,000 in decimal notation.

K: When referring to storage capacity, 2 to the tenth power; 1,024 in decimal notation.

KB: Kilobyte; 1,024 bytes.

kHz: A unit of frequency equal to 1,000 hertz.

kilo (k): One thousand.

latch: (1) A feedback loop in symmetrical digital circuits used to maintain a state. (2) A simple logic-circuit storage element comprising two gates as a unit.

LED: Light-emitting diode.

light-emitting diode (LED): A semi-conductor chip that gives off visible or infrared light when activated.

low-order position: The rightmost position in a string of characters.

m: (1) Milli; one thousand or thousandth part. (2) Meter.

M: Mega; 1,000,000 in decimal notation. When referring to storage capacity, 2 to the twentieth power; 1,048,576 in decimal notation.

mA: Milliampere.

machine language: (1) A language that is used directly by a machine. (2) Another term for computer instruction code.

main storage: A storage device in which the access time is effectively independent of the location of the data.

MB: Megabyte, 1,048,576 bytes.

mega (M): 10 to the sixth power, 1,000,000 in decimal notation. When referring to storage capacity, 2 to the twentieth power, 1,048,576 in decimal notation.

megabyte (MB): 1,048,576 bytes.

megahertz (MHz): A unit of measure of frequency. 1 megahertz equals 1,000,000 hertz.

MFM: Modified frequency modulation.

MHz: Megahertz.

microprocessor: An integrated circuit that accepts coded instructions for execution; the instructions may be entered, integrated, or stored internally.

microsecond (μs): One-millionth of a second.

milli (m): One thousand or one thousandth.

milliampere (mA): One thousandth of an ampere.

millisecond (ms): One thousandth of a second.

mnemonic: A symbol chosen to assist the human memory; for example, an abbreviation such a "mpy" for "multiply."

mode: (1) A method of operation; for example, the binary mode, the interpretive mode, the alphanumeric mode. (2) The most frequency value in the statistical sense.

modem: (Modulator-Demodulator) A device that converts serial (bit by bit) digital signals from a business machine (or data terminal equipment) to analog signals which are suitable for transmission in a telephone network. The inverse function is also performed by the modem on reception of analog signals.

modified frequency modulation (MFM): The process of varying the amplitude and frequency of the "write" signal. MFM pertains to the number of bytes of storage that can be stored on the recording media. The number of bytes is twice the number contained in the same unit area of recording media at single density.

modulo check: A calculation performed on values entered into a system. This calculation is designed to detect errors.

monitor: (1) A device that observes and verifies the operation of a data processing system and indicates any specific departure from the norm. (2) A television type display, such as the IBM Monochrome Display. (3) Software or hardware that observes, supervises, controls, or verifies the operations of a system.

ms: Millisecond; one thousandth of a second.

multiplexer: A device capable of interleaving the events of two or more activities, or capable of distributing the events of an interleaved sequence to the respective activities.

NAND: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,...,then the NAND of P,Q,R,...is true if at least one statement is false, false if all statements are true.

nanosecond (ns): One-thousandth-millionth of a second.

nonconjunction: The dyadic boolean operation the result of which has the boolean value 0 if, and only if, each operand has the boolean value 1.

non-return-to-zero inverted (NRZI): A transmission encoding method in which the data terminal equipment changes the signal to the opposite state to send a binary 0 and leaves it in the same state to send a binary 1.

NOR: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,...,then the NOR of P,Q,R,...is true if all statements are false, false if at least one statement is true.

NOT: A logical operator having the property that if P is a statement, then the NOT of P is true if P is false, false if P is true.

NRZI: Non-return-to-zero inverted.

ns: Nanosecond; one-thousandth-millionth of a second.

operating system: Software that controls the execution of programs; an operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

OR: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,...,then the OR of P,Q,R,...is true if at least one statement is true, false if all statements are false.

output: Pertaining to a device, process, or channel involved in an output process, or to the data or states involved in an output process.

output process: (1) The process that consists of the delivery of data from a data processing system, or from any part of it. (2) The return of information from a data processing system to an end user, including the translation of data from a machine language to a language that the end user can understand.

overcurrent: A current of higher than specified strength.

overvoltage: A voltage of higher than specified value.

parallel: (1) Pertaining to the concurrent or simultaneous operation of two or more devices, or to the concurrent performance of two or more activities. (2) Pertaining to the concurrent or simultaneous occurrence of two or more related activities in multiple devices or channels. (3) Pertaining to the simultaneity of two or more processes. (4) Pertaining to the simultaneous processing of the individual parts of a whole, such as the bits of a character and the characters of a word, using separate facilities for the various parts. (5) Contrast with serial.

PEL: Picture element.

personal computer: A small home or business computer that has a processor and keyboard that can be connected to a television or some other monitor. An optional printer is usually available.

picture element (PEL): (1) The smallest displayable unit on a display. (2) Synonymous with pixel, PEL.

pinout: A diagram of functioning pins on a pinboard.

pixel: Picture element.

polling: (1) Interrogation of devices for purposes such as to avoid contention, to determine operational status, or to determine readiness to send or receive data. (2) The process whereby stations are invited, one at a time, to transmit.

port: An access point for data entry or exit.

printed circuit board: A piece of material, usually fiberglass, that contains a layer of conductive material, usually metal. Miniature electronic components on the fiberglass transmit electronic signals through the board by way of the metal layers.

program: (1) A series of actions designed to achieve a certain result. (2) A series of instructions telling the computer how to handle a problem or task. (3) To design, write, and test computer programs.

programming language: (1) An artificial language established for expressing computer programs. (2) A set of characters and rules, with meanings assigned prior to their use, for writing computer programs.

PROM: Programmable read-only memory.

propagation delay: The time necessary for a signal to travel from one point on a circuit to another.

radix: (1) In a radix numeration system, the positive integer by which the weight of the digit place is multiplied to obtain the weight of the digit place with the next higher weight; for example, in the decimal numeration system, the radix of each digit place is 10. (2) Another term for base.

radix numeration system: A positional representation system in which the ratio of the weight of any one digit place to the weight of the digit place with the next lower weight is a positive integer. The permissible values of the character in any digit place range from zero to one less than the radix of the digit place.

RAS: Row address strobe.

RGBI: Red-green-blue-intensity.

read-only memory (ROM): A storage device whose contents cannot be modified, except by a particular user, or when operating under particular conditions; for example, a storage device in which writing is prevented by a lockout.

read/write memory: A storage device whose contents can be modified.

red-green-blue-intensity (RGBI): The description of a direct-drive color monitor which accepts red, green, blue, and intensity signal inputs.

register: (1) A storage device, having a specified storage capacity such as a bit, a byte, or a computer word, and usually intended for a special purpose. (2) On a calculator, a storage device in which specific data is stored.

RF modulator: The device used to convert the composite video signal to the antenna level input of a home TV.

ROM: Read-only memory.

ROM/BIOS: The ROM resident basic input/output system, which provides the device level control of the major I/O devices in the computer system.

row address strobe (RAS): A signal that latches the row addresses in a memory chip.

RS-232C: The standard set by the EIA for communications between computers and external equipment.

RTS: Request to send. Associated with modem control.

run: A single continuous performance of a computer program or routine.

scan line: The use of a cathode beam to test the cathode ray tube of a display used with a personal computer.

schematic: The description, usually in diagram form, of the logical and physical structure of an entire data base according to a conceptual model.

SDLC: Synchronous Data Link Control.

sector: That part of a track or band on a magnetic drum, a magnetic disk, or a disk pack that can be accessed by the magnetic heads in the course of a predetermined rotational displacement of the particular device.

serdes: Serializer/deserializer.

serial: (1) Pertaining to the sequential performance of two or more activities in a single device. In English, the modifiers serial and parallel usually refer to devices, as opposed to sequential and consecutive, which refer to processes. (2) Pertaining to the sequential or consecutive occurrence of two or more related activities in a single device or channel. (3) Pertaining to the sequential processing of the individual parts of a whole, such as the bits of a character or the characters of a word, using the same facilities for successive parts. (4) Contrast with parallel.

sink: A device or circuit into which current drains.

software: (1) Computer programs, procedures, rules, and possibly associated documentation concerned with the operation of a data processing system. (2) Contrast with hardware.

source: The origin of a signal or electrical energy.

source circuit: (1) Generator circuit. (2) Control with sink.

SS: Start-stop transmission.

start bit: Synonym for start signal.

start-of-text character (STX): A transmission control character that precedes a text and may be used to terminate the message heading.

start signal: (1) A signal to a receiving mechanism to get ready to receive data or perform a function. (2) In a start-stop system, a signal preceding a character or block that prepares the receiving device for the reception of the code elements. Synonymous with start bit.

start-stop (SS) transmission: Asynchronous transmission such that a group of signals representing a character is preceded by a start signal and followed by a stop signal. (2) Asynchronous transmission in which a group of bits is preceded by a start bit that prepares the receiving mechanism for the reception and registration of a character and is followed by at least one stop bit that enables the receiving mechanism to come to an idle condition pending the reception of the next character.

stop bit: Synonym for stop signal.

stop signal: (1) A signal to a receiving mechanism to wait for the next signal. (2) In a start-stop system, a signal following a character or block that prepares the receiving device for the reception of a subsequent character or block. Synonymous with stop bit.

strobe: (1) An instrument used to determine the exact speed of circular or cyclic movement. (2) A flashing signal displaying an exact event.

STX: Start-of-text character.

Synchronous Data Link Control (SLDC): A protocol for the management of data transfer over a data communications link.

synchronous transmission: Data transmission in which the sending and receiving devices are operating continuously at the same frequency and are maintained, by means of correction, in a desired phase relationship.

text: In ASCII and data communication, a sequence of characters treated as an entity if preceded and terminated by one STX and one ETX transmission control, respectively.

track: (1) The path or one of the set of paths, parallel to the reference edge on a data medium, associated with a single reading or writing component as the data medium moves past the component. (2) The portion of a moving data medium such as a drum, tape, or disk, that is accessible to a given reading head position.

transistor-transistor logic (TTL): A circuit in which the multiple-diode cluster of the diode-transistor logic circuit has been replaced by a multiple-emitter transistor.

TTL: Transistor-transistor logic.

TX Data: Transmit data. Associated with modem control. External connections of the RS-232C asynchronous communications adapter interface.

video: Computer data or graphics displayed on a cathode ray tube, monitor or display.

write precompensation: The varying of the timing of the head current from the outer tracks to the inner tracks of the diskette to keep a constant write signal.

BIBLIOGRAPHY

Intel Corporation. *The 8086 Family User's Manual*
This manual introduces the 8086 family of microcomputing components and serves as a reference in system design and implementation.

Intel Corporation. *8086/8087/8088 Macro Assembly Reference Manual for 8088/8085 Based Development System*
This manual describes the 8086/8087/8088 Macro Assembly Language, and is intended for use by persons who are familiar with assembly language.

Intel Corporation.*Component Data Catalog*
This book describes Intel components and their technical specifications.

Motorola, Inc.*The Complete Microcomputer Data Library.*
This book describes Motorola components and their technical specificaitons.

National Semiconductor Corporation. *INS 8250 Asynchronous Communications Element.* This book documents physical and operating characteristics of the INS 8250.

Notes:

INDEX

A

A/N mode (alphanumeric mode) 1-131
A0-A19 (Address Bits 0 to 19), I/O channel 1-18
adapter card with ROM 2-10
adapter,
  asynchronous communication 1-223
  binary synchronous communication 1-251
  color/graphics monitor 1-131
  diskette drive 1-159
  fixed disk drive 1-187
  game control 1-211
  monochrome display and printer 1-129
  printer 1-117
  synchronous data link control 1-271
Address Bits 0 to 19 (A0-A19), I/O channel 1-131
Address Bits (asynchronous communication) 1-225
Address Enable (AEN), I/O channel 1-22
Address Latch Enable (ALE), I/O channel 1-20
address map, I/O 1-10
AEN (Address Enable), I/O channel 1-22
ALE (Address Latch Enable), I/O channel 1-20
all points addressable mode 1-129, 1-132
alphanumeric mode, 1-136
  high resolution 1-137
  low resolution 1-137
alt (keyboard extended code) 2-15
APA mode (all points addressable mode) 1-131, 1-132
asynchronous communications adapter, 1-223
  adapter address jumper module 1-249
  address bits 1-225
  block diagram 1-224
  connector specifications 1-250
  current loop interface 1-227
  divisor latch least significant bit 1-237
  divisor latch most significant bit 1-238
  I/O decode 1-225
  INS8250 functional pin description 1-229
  INS8250 input signals 1-229
  INS8250 input/output signals 1-233
  INS8250 output signals 1-232
  interface descriptions 1-226
  interface format jumper module 1-249

Index

interrupt control functions 1-237
interrupt enable register 1-243
interrupt identification register 1-240
interrupts 1-226
line control register 1-235
line status register 1-239
modem control register 1-244
modem status register 1-246
modes of operation 1-224
programmable baud rate generator 1-237
programming considerations 1-234
receiver buffer register 1-247
reset functions 1-230
transmitter holding register 1-248
voltage interchange information 1-228
attributes, character
(see character attributes)

B

BASIC reserved interrupts 2-7
BASIC,
DEF SEG 2-8
reserved interrupt 2-7
screen editor keyboard functions 2-20
workspace variables 2-8
baud rate generator 1-237
bell (printer) 1-102
bibliography I-1
binary synchronous communications adapter, 1-251
8251A programming procedures 1-262
8251A universal synchronous/asynchronous receiver/transmitter 1-252
8253-5 programmable interval timer 1-257
8255A-5 programmable peripheral interface 1-256
block diagram 1-252
command instruction format 1-264
connector information 1-269
data bus buffer 1-253
interface signal information 1-266
interrupt information 1-268
mode instruction definition 1-263
read/write control logic 1-253
receive 1-258

receiver buffer 1-255
receiver control 1-255
status read definition 1-265
transmit 1-265
transmitter buffer 1-254
transmitter control 1-255
typical programming sequence 1-259
BIOS,
cassette logic (see cassette logic BIOS)
fixed disk ROM A-87
memory map 2-9
parameter passing 2-3
software interrupt listing 2-4
system ROM A-2
use of 2-2
bisync communications
(see binary synchronous communications)
block diagram
8251A universal synchronous/asynchronous receiver/transmitter 1-252
8273 SDLC protocol controller 1-272
asynchronous communications adapter 1-224
cassette circuits 1-29
color/graphics monitor adapter 1-134
coprocessor 1-37
diskette drive adapter 1-160
expansion board 1-80
extender card 1-87
fixed disk drive adapter 1-188
game control adapter 1-211
keyboard interface 1-75
monochrome display adapter 1-124
printer adapter 1-118
prototype card 1-218
receiver card 1-89
speaker drive system 1-24
synchronous data link control adapter 1-271
system 1-2
break (keyboard extended code) 2-17
BSC adapter
(see binary synchronous communications)

Index

C

cable
  communications adapter 1-301
  expansion unit 1-79
  printer 1-91
cancel (printer) 1-103
cancel ignore paper end (printer) 1-103
cancel skip perforation (printer) 1-109
caps lock (keyboard extended code) 2-16
card dimensions and specifications E-4
card,
  dimensions and specifications E-4
  extender 1-85
  prototype 1-217
  receiver 1-88
carriage return (printer) 1-102
cassette circuit block diagram
  motor control 1-30
  read hardware 1-29
  write hardware 1-29
cassette interface, 1-28
  connector specifications 1-31
cassette logic,
  BIOS 2-21
  cassette read 2-23
  cassette write 2-22
  data record architecture 2-24
  data record components 2-24
  error recovery 2-24
  interrupt 2-21
  software algorithms 2-28
cassette read 2-23
cassette ROM BIOS 2-21
cassette write 2-22
CCITT, F-1
  standards F-1
character attributes
  color/graphics monitor adapter 1-140
  monochrome display adapter 1-140
character codes
  keyboard 2-11

character set,
graphics printer (set 1) 1-113
graphics printer (set 2) 1-115
matrix printer 1-111
quick reference C-12
clear printer buffer (printer) 1-110
CLK (system clock), I/O channel 1-19
color display 1-157
operating characteristics 1-157
specifications E-2
color select register 1-149
color/graphics monitor adapter 1-131
6845 register description 1-148
alphanumeric mode 1-136
alphanumeric mode (high-resolution) 1-144
alphanumeric mode (low-resolution) 1-142
block diagram 1-134
character attributes 1-140
color-select register 1-149
composite connector specifications 1-155
connector specifications 1-156
direct-drive connector specifications 1-155
display buffer basic operation 1-145
graphics mode 1-141
graphics mode (high resolution) 1-144
graphics mode (low resolution) 1-142
graphic mode (medium resolution) 1-142
light pen connector specifications 1-156
major components 1-135
memory requirements 1-154
mode control and status register 1-149
mode register summary 1-152
mode select register 1-151
programming considerations 1-147
RF modulator connector specifications 1-156
sequence of events 1-153
status register 1-153
summary of available colors 1-146
colors, summary of available 1-146
command status register 0 1-172
command status register 1 1-173
command status register 2 1-174
command status register 3 1-175

command summary,
diskette drive adapter 1-166
fixed disk drive adapter 1-195
communications adapter cable 1-301
connector specifications 1-302
communications F-1
establishing a link F-3
component diagram,
system board 1-7
compressed (printer) 1-103
compressed off (printer) 1-103
connector specifications,
asynchronous communications adapter 1-250
binary synchronous communications 1-269
cassette interface 1-31
color/graphics monitor adapter 1-155
communications adapter cable 1-302
diskette drive adapter (external) 1-182
diskette drive adapter (internal) 1-181
game control adapter 1-216
keyboard interface 1-78
monochrome display adapter 1-128
printer adapter 1-122
synchronous data link control adapter 1-299
connectors,
power supply (system unit) 1-26
power supply (expansion unit) 1-83
considerations, programming
(see programming considerations)
control byte, fixed disk drive adapter 1-194
control codes, printer 1-101
control/read/write logic 1-274
coprocessor,
(see math coprocessor)
ctrl (keyboard extended code) 2-15
current loop interface 1-227

D

D0-D7 (data bits 0 to 7), I/O channel 1-18
DACK0-DACK3 (DMA Acknowledge 0 to 3), I/O channel 1-21
Data Bits 0 to 7 (D0-D7), I/O channel 1-18
data flow,
system board 1-8

data record architecture, cassette 2-24
data record components, cassette 2-24
data register 1-193
data transfer mode register 1-288
DEF SEG (default segment workspace) 2-8
default workspace segment (DEF SEG) 2-8
diagram, block (see block diagram)
digital output register 1-161
diskette drive adapter 1-159
  adapter input 1-179
  adapter output 1-178
  block diagram 1-160
  command status register 0 1-172
  command status register 1 1-173
  command status register 2 1-174
  command status register 3 1-175
  command summary 1-166
  connector specifications (external) 1-182
  connector specifications (internal) 1-181
  digital-output register 1-161
  DPC registers 1-174
  drive A and B interface 1-178
  drive constants 1-176
  FDC constants 1-176
  floppy disk controller 1-162
  functional description 1-161
  programming considerations 1-164
  programming summary 1-175
  symbol descriptions 1-164
  system I/O channel interface 1-176
diskette drive, 1-183
  electrical specifications 1-184
  mechanical specifications 1-184
  switch settings G-1
diskettes 1-185
display adapter type switch settings G-1
display,
  color 1-157
  monochrome 1-123
divisor latch,
  least significant bit 1-237
  most significant bit 1-238
DMA Acknowledge 0 to 3 (DACK0-DACK3),
 I/O channel 1-21

Index

DMA Request 1 to 3 (DRQ1-DRQ3), I/O channel 1-21
DOS reserved interrupts 2-9
DOS,
  keyboard functions 2-21
  reserved interrupts 2-9
double strike (printer) 1-106
double strike off (printer) 1-107
double width (printer) 1-99, 1-103
double width off (printer) 1-103
DPC registers 1-175
DRQ1-DRQ3 (DMA Request 1 to 3), I/O channel 1-21

E

EIA, F-1
  standards F-1
emphasized (printer) 1-106
emphasized off (printer) 1-106
error recovery, cassette 2-24
escape (printer) 1-104
establishing a communications link F-3
expansion board, 1-79
  block diagram 1-80
expansion channel 1-81
expansion unit, 1-79
  cable 1-79
  expansion board 1-79
  expansion channel 1-81
  extender card 1-85
  interface information 1-90
  power supply 1-83
  power supply connectors 1-83
  receiver card 1-88
  specifications E-2
extender card, 1-85
  block diagram 1-87
  programming considerations 1-86
  switch settings G-1

F

FABS 1-44
FADD 1-43
FBLD 1-45
FBSTP 1-46
FCHS 1-46
FCLEX/FNCLEX 1-46
FCOM 1-47
FCOMP 1-47
FCOMPP 1-48
FDECSTP 1-48
FDISI/FNDISI 1-48
FDIV 1-49
FDIVR 1-50
FENI/FNENI 1-51
FFREE 1-51
FICOM 1-51
FICOMP 1-52
FILD 1-52
FINCSTP 1-52
FINIT/FNINIT 1-53
FIST 1-54
FISTP 1-54
fixed disk controller 1-185
fixed disk drive 1-201
fixed disk drive adapter 1-185
  block diagram 1-186
  command summary 1-193
  control byte 1-192
  data register 1-191
  fixed disk controller 1-185
  interface specifications 1-200
  programming considerations 1-187
  programming summary 1-197
  ROM BIOS listing A-87
  sense bytes 1-187
  status register 1-187
  system I/O channel interface 1-198
fixed disk drive, 1-201
  electrical specifications 1-202
  mechanical specifications 1-202

fixed disk ROM BIOS A-87
FLD 1-55
FLDCW 1-55
FLDENV 1-56
FLDLG2 1-56
FLDLN2 1-56
FLDL2E 1-57
FLDL2T 1-57
FLDPI 1-57
FLDZ 1-58
FLD1 1-58
floppy disk controller 1-160
FMUL 1-59
FNOP 1-60
FPATAN 1-60
FPREM 1-60
FPTAN 1-61
FRNDINT 1-61
FRSTOR 1-61
form feed (printer) 1-102
FSAVE/FNSAVE 1-62
FSCALE 1-62
FSQRT 1-62
FST 1-63
FSTCW/FNSTCW 1-63
FSTENV/FNSTENV 1-64
FSTP 1-64
FSTSW/FNSTSW 1-65
FSUB 1-65
FSUBR 1-66
FTST 1-67
FWAIT 1-68
FXAM 1-68
FXCH 1-69
FXTRACT 1-70
FYL2X 1-70
FYL2XP1 1-71
F2XM1 1-71

G

game control adapter, 1-211
  block diagram 1-211
  connector specifications 1-216

functional description 1-212
I/O channel description 1-213
interface description 1-214
joy stick schematic diagram 1-215
glossary, H-1
graphics mode, 1-141
high resolution 1-144
low resolution 1-142
medium resolution 1-142

H

hardware interrupt listing 1-11
home head (printer) 1-105
horizontal tab (printer) 1-102

I

I/O address map 1-10
I/O bit map, 8255A 1-12
I/O CH CK (I/O Channel Check), I/O channel 1-20
I/O CH RDY (I/O Channel Ready), I/O channel 1-20
I/O Channel Check (I/O CH CK), I/O channel 1-20
I/O channel interface,
diskette drive adapter 1-176
fixed disk drive adapter 1-187
prototype card 1-217
I/O Channel Ready (I/O CH RDY), I/O channel 1-20
I/O channel, 1-17
-I/O Channel Check (I/O CH CK) 1-20
-I/O Read Command (IOR) 1-20
-I/O Write Command (IOW) 1-21
Address Bits 0 to 19 (A0-A19) 1-20
Address Enable (AEN) 1-21
Address Latch Enable (ALE) 1-20
Data Bits 0 to 7 (D0-D7) 1-20
description 1-20
diagram 1-18
DMA Request 1 to 3 (DRQ1-DRQ3) 1-21
I/O Channel Ready (I/O CH RDY) 1-20
Interrupt Request 2 to 7 (IRQ2-IRQ7) 1-20
Memory Read Command (MEMR) 1-21

Memory Write Command (MEMW) 1-21
Oscillator (OSC) 1-19
Reset Drive (RESET DRV) 1-20
System Clock (CLK) 1-20
Terminal Count (T/C) 1-22
I/O Read Command (IOR), I/O channel 1-21
I/O Write Command (IOW), I/O channel 1-21
IBM 10MB Fixed Disk Drive 1-201
IBM 5-1/4" Diskette Drive 1-183
IBM 5-1/4" Diskette Drive Adapter 1-159
IBM 80 CPS Graphics Printer 1-91
IBM 80 CPS Matrix Printer 1-91
IBM 80 CPS Printers 1-91
IBM Asynchronous Communications Adapter 1-223
IBM Binary Synchronous Communications Adapter 1-251
IBM Color Display 1-157
IBM Color/Graphics Monitor Adapter 1-131
IBM Communicatons Adapter Cable 1-301
IBM Fixed Disk Drive Adapter 1-187
IBM Game Control Adapter 1-211
IBM Memory Expansion Options 1-205
IBM Monochrome Display and Printer Adapter 1-223
IBM Monochrome Display 1-129
IBM Personal Computer Math Coprocessor 1-33
IBM Printer Adapter 1-117
IBM Prototype Card 1-215
IBM Synchronous Data Link Controller Adapter 1-271
ignore paper end (printer) 1-104
INS8250,
(see National Semiconductor INS8250)
Intel 8088 microprocessor,
arithmetic B-7
conditional transfer operations B-14
control transfer B-11
data transfer B-5
hardware interrupt listing 1-8
instruction set index B-18
instruction set matrix B-16
logic B-9
memory segmentation model B-4
operand summary B-15
processor control B-15
register model B-2

second instruction byte summary B-3
segment override prefix B-4
software interrupt listing 2-4
string manipulation B-10
use of segment override B-4
Intel 8253-5 Programmable Interval Timer
(see synchronous data link control communications adapter)
Intel 8255A Programmable Peripheral Interface
I/O bit map 1-12
Intel 8255A-5 Programmable Peripheral Interface
(see synchronous data link control communications adapter)
Intel 8273 SDLC Protocol Controller
(see synchronous data link control communications adapter)
block diagram 1-273
interrupt enable register 1-243
interrupt identification register 1-243
interrupt listing,
8088 hardware 1-11
8088 software 2-4
Interrupt Request 1 to 7 (IRQ2-IRQ7), I/O channel 1-20
interrupts,
8088 hardware 1-11
8088 software 2-4
asynchronous communications adapter 1-223
BASIC reserved 2-7
DOS reserved 2-21
special 2-7
IOR (I/O Read Command), I/O channel 1-20
IOW (I/O Write Command), I/O channel 1-21
IRQ2-IRQ7 (Interrupt Request 2 to 7), I/O channel 1-20

J

joy stick,
positions 1-221
schematic diagram 1-215
jumper module, asynchronous communications adapter 1-249

K

keyboard extended codes,
- alt 2-15
- break 2-16
- caps lock 2-16
- ctrl 2-15
- pause 2-17
- print screen 2-17
- scroll lock 2-16
- shift 2-15
- shift key priorities 2-16
- shift states 2-15
- system reset 2-16

keyboard 1-73
- BASIC screen editor special functions 2-20
- character codes 2-11
- commonly used functions 2-18
- diagram 1-76
- DOS special functions 2-20
- encoding 2-11
- extended functions 2-14
- interface block diagram 1-75
- interface connector specifications 1-78
- scan codes 1-77
- specifications E-1

L

light pen connector specifications 1-156
line control register 1-235
line feed (printer) 1-102
line status register 1-239
logic diagrams D-1

M

math coprocessor 1-33
- block diagram 1-37
- control unit 1-37
- control word 1-40

data types 1-34
exception pointers 1-41
FABS 1-44
FADD 1-44
FBLD 1-45
FBSTP 1-46
FCHS 1-46
FCLEX/FNCLEX 1-46
FCOM 1-47
FCOMP 1-47
FCOMPP 1-48
FDECSTP 1-48
FDISI/FNDISI 1-48
FDIV 1-49
FDIVR 1-50
FENI/FNENI 1-51
FFREE 1-51
FICOM 1-51
FICOMP 1-52
FILD 1-52
FINCSTP 1-52
FINIT/FNINIT 1-53
FIST 1-54
FISTP 1-54
FLD 1-55
FLDCW 1-55
FLDENV 1-56
FLDLG2 1-56
FLDLN2 1-56
FLDL2E 1-57
FLDL2T 1-57
FLDPI 1-57
FLDZ 1-58
FLD1 1-58
FMUL 1-59
FNOP 1-60
FPATAN 1-60
FPREM 1-60
FPTAN 1-61
FRNDINT 1-61
FRSTOR 1-61

FSAVE/FNSAVE 1-62
FSCALE 1-62
FSQRT 1-62
FST 1-63
FSTCW/FNSTCW 1-63
FSTENV/FNSTENV 1-64
FSTP 1-64
FSTSW/FNSTSW 1-65
FSUB 1-65
FSUBR 1-66
FTST 1-67
FWAIT 1-68
FXAM 1-68
FXCH 1-69
FXTRACT 1-70
FYL2X 1-70
FYL2XP1 1-71
F2XM1 1-71
hardware interface 1-35
instruction set 1-43
interconnection 1-36
number system 1-42
programming interface 1-34
register stack 1-38
status word 1-39
tag word 1-41
memory expansion options, 1-205
DIP module start address 1-208
memory module description 1-206
memory module pin configuration 1-207
memory option switch settings G-1
R/W memory operating characteristics 1-206
switch-configurable start address 1-208
memory locations,
reserved 2-8
memory map,
BIOS 2-9
system 1-13
Memory Read Command (MEMR), I/O channel 1-21
memory switch settings, G-1
extender card G-1
memory options G-1
system board G-1

Memory Write Command (MEMW), I/O channel 1-21
(MEMR) Memory Read Command, I/O channel 1-21
(MEMW) Memory Write Command, I/O channel 1-21
microprocessor (see Intel 8088 microprocessor)
mode control and status register 1-149
mode select register 1-151
modem control register 1-244
modem status register 1-246
monochrome display 1-129
monochrome display and printer adapter 1-123
monochrome display adapter 1-123
  6845 CRT control port 1-127
  6845 CRT status port 1-127
  block diagram 1-124
  character attributes 1-138
  connector specifications 1-128
  I/O address and bit map 1-127
  programming considerations 1-125
monochrome display, 1-129
  operating characteristics 1-129
  specifications E-3
Motorola 6845 CRT Controller,
  (see color/graphics monitor adapter)
  (see monochrome display adapter)

N

National Semiconductor INS8250 Asynchronous
  (see asynchronous communications adapter)
  functional pin description 1-229
  input signals 1-229
  input/output signals 1-233
  output signals 1-232
null (printer) 1-102

O

one bit delay mode register 1-289
operating mode register 1-289
OSC (oscillator) 1-19
Oscillator (OSC), I/O channel 1-19
over-voltage/over-current (expansion unit) 1-84
over-voltage/over-current (system unit) 1-27

P

parameter passing (ROM BIOS) 2-3
pause (keyboard extended code) 2-17
power good signal (expansion unit) 1-84
power good signal (system unit) 1-27
power supply (expansion unit) 1-82
  connectors 1-83
  input requirements 1-82
  over-voltage/current protection 1-84
  pin assignments 1-83
  power good signal 1-83
  Vac output 1-82
  Vdc output 1-82
power supply (system unit) 1-23
  connectors and pin assignments 1-26
  input requirements 1-24
  over-voltage/current protection 1-27
  pin assignments 1-26
  power good signal 1-27
  Vac output 1-25
  Vdc output 1-25
print screen (keyboard extended code) 2-17
printer adapter, 1-117
  block diagram 1-118
  connector specifications 1-122
  programming considerations 1-119
printer control codes, 1-101
  1/8-inch line feeding 1-104
  1920 bit-image graphics mode 1-110
  480 bit-image graphics mode 1-107
  7/72-inch line feeding 1-104
  960 bit-image graphics mode 1-109
  960 bit-image graphics mode normal speed 1-110
  bell 1-102
  cancel 1-103
  cancel ignore paper end 1-105
  cancel skip perforation 1-109
  carriage return 1-102
  clear printer buffer 1-110
  compressed 1-103
  compressed off 1-103
  double strike 1-106
  double strike off 1-107
  double width 1-103, 1-110

double width off 1-103
emphasized 1-106
emphasized off 1-106
escape 1-103
form feed 1-102
home head 1-105
horizontal tab 1-102
ignore paper end 1-104
line feed 1-102
null 1-102
printer deselected 1-103
printer selected 1-103
select character set 1 1-104
select character set 2 1-104
set horizontal tab stops 1-106
set lines per page 1-106
set skip perforation 1-109
set variable line feeding 1-105, 1-107
set vertical tabs 1-105
starts variable line feeding 1-105
subscript/superscript 1-109
subscript/superscript off 1-109
underline 1-104
unidirectional printing 1-109
vertical tab 1-102
printer deselected (printer) 1-103
printer selected (printer) 1-103
printer, 1-91
additional specifications 1-93
cable 1-91
connector pin assignment 1-97
control codes 1-101
graphic character set 1 1-113
graphic character set 2 1-115
interface signal descriptions 1-96
matrix character set 1-111
modes 1-100
parallel interface 1-96
parallel interface timing diagram 1-96
specifications 1-92, E-3
switch locations 1-94
switch settings 1-94
processor (see Intel 8088 micrprocessor)
programmable baud rate generator 1-237

programming considerations,
asynchronous communications adapter 1-234
binary synchronous communications adapter 1-259
color/graphics monitor adapter 1-131
diskette drive adapter 1-159
extender card 1-86
fixed disk drive adapter 1-187
monochrome display adapter 1-123
printer adapter 1-123
receiver card 1-88
SDLC adapter 1-281
prototype card, 1-217
block diagram 1-218
external interface 1-222
I/O channel interface 1-219
layout 1-219
system loading and power limitations 1-221

Q

quick reference, character set C-12

R

receiver buffer register 1-247
receiver card, 1-88
block diagram 1-89
programming considerations 1-86
register,
6845 description (color/graphic adapter) 1-146
color select (color/graphic adapter) 1-147
command status 0 (diskette drive adapter) 1-172
command status 1 (diskette drive adapter) 1-173
command status 2 (diskette drive adapter) 1-174
command status 3 (diskette drive adapter) 1-175
data (fixed disk drive adapter) 1-192
data transfer mode (SDLC) 1-288
digital output (diskette drive adapter) 1-161
DPC (diskette drive adapter) 1-175
interrupt enable (asynchronous communications) 1-243
interrupt identification (asynchronous communications) 1-241
line control (asynchronous communications) 1-234
line status (asynchronous communications) 1-239
mode control and status (color/graphics) 1-149

mode select (color/graphics) 1-151
modem control (asynchronous communications) 1-244
modem status (asynchronous communications) 1-246
one-bit delay mode (SDLC) 1-289
operating mode (SDLC) 1-286
receiver buffer (asynchronous communications) 1-247
serial I/O mode (SDLC) 1-288
status (color/graphics) 1-153
status (fixed disk drive adapter) 1-187
transmitter holding (asynchronous communications) 1-248
reserved interrupts,
BASIC and DOS 2-7
reserved memory locations 2-7
Reset Drive (RESET DRV), I/O channel 1-19
RESET DRV (Reset Drive), I/O channel 1-19
RF modulator connector specifications 1-156
ROM BIOS, 2-2
Cassette A-74
Fixed Disk A-87
System A-2
ROM, adapter cards with 2-10
RS-232C,
interface standards F-2

S

scan codes,
keyboard 1-77
scroll lock (keyboard extended code) 2-16
SDLC (see synchronous data link control)
select character set 1 (printer) 1-104
select character set 2 (printer) 1-104
sense bytes, fixed disk drive adapter 1-189
serial I/O mode register 1-288
set horizontal tab stops (printer) 1-106
set lines per page (printer) 1-106
set skip perforation (printer) 1-109
set variable line feeding (printer) 1-105, 1-107
set vertical tabs (printer) 1-105
shift (keyboard extended code) 2-15
shift key priorities (keyboard code) 2-16
shift states (keyboard extended code) 2-15
software interrupt listing 2-4
speaker connector 1-23
speaker drive system 1-23
speaker interface 1-23

Index

specifications,
80 CPS printers E-3
color display E-2
expansion unit E-2
keyboard E-1
monochrome display E-3
printer 1-92
printer (additional) 1-93
system unit E-1
stack area 2-7
starts variable line feeding (printer) 1-104
status register,
color/graphics monitor adapter 1-154
fixed disk drive adapter 1-189
synchronous data link control adapter 1-282
subscript/superscript (printer) 1-109
subscript/superscript off (printer) 1-109
switch settings, G-1
diskette drive G-1
display adapter type G-1
extender card G-1
memory options G-1
printer 1-91
system board G-1
system board memory G-1
synchronous data link control communications adapter, 1-271
8253-5 interval timer control word 1-285
8253-5 progammable interval timer 1-281
8255A-5 port A assignments 1-280
8255A-5 port B assignments 1-280
8255A-5 port C assignments 1-281
8255A-5 programmable peripheral interface 1-280
8273 command phase flow chart 1-292
8273 commands 1-291
8273 control/read/write registers 1-275
8273 data interfaces 1-276
8273 elements of data transfer interface 1-276
8273 mode register commands 1-288
8273 modem control block 1-277
8273 modem control port A 1-277
8273 modem control port B 1-278
8273 modem interface 1-277
8273 protocol controller operations 1-271
8273 protocol controller structure 1-273
8273 register selection 1-274
8273 SDLC protocol controller block diagram 1-273

8273 transmit/receiver timing 1-279
block diagram 1-271
command phase 1-290
connector specifications 1-299
control/read/write logic 1-274
data transfer mode register 1-288
device addresses 1-297
execution phase 1-293
general receive 1-294
initialization/configuration commands 1-286
initializing the SDLC adapter 1-283
interface information 1-298
interrupt information 1-297
one bit delay code register 1-289
operating mode register 1-286
partial byte received codes 1-296
processor interface 1-274
programming considerations 1-281
protocol control module features 1-272
protocol controller operations 1-272
result code summary 1-296
result phase 1-291
selective receive 1-295
serial data timing block 1-279
serial I/O mode register 1-288
status register format 1-282
transmit 1-294
system block diagram 1-2
system board, 1-3
component diagram 1-7
data flow 1-8
R/W memory operating characteristics 1-202
switch settings G-1
System Clock (CLK), I/O channel 1-19
system memory map 1-13
system reset (keyboard extended code) 2-16
system ROM BIOS A-2
system unit, 1-3
cassette interface 1-31
I/O channel 1-17
I/O channel diagram 1-18
keyboard interface 1-78
power supply 1-23
speaker interface 1-22
specifications E-1
system board 1-3

T

T/C (Terminal Count), I/O channel 1-22
transmitter holding register 1-248

U

underline (printer) 1-104
unidirectional printer (printer) 1-109

V

Vac output,
  expansion unit 1-82
  system unit 1-25
Vdc output,
  expansion unit 1-82
  system unit 1-25
vectors with special meanings 2-5
vertical tab (printer) 1-102
voltage interchange,
  asynchronous communications adapter 1-228

Numerics

1/8 inch line feeding (printer) 1-104
1920 bit-image graphics mode (printer) 1-110
480 bit-image graphics mode (printer) 1-107
6845,
  (see color/graphics monitor adapter)
  (see monochrome display adapter)
7/72 inch line feeding (printer) 1-104
8088,
  (see Intel 8088 microprocessor)
8250,
  (see asynchronous communications adapter)
8253-5,
  (see synchronous data link control adapter)
8255A 1-12
8255A-5,
  (see synchronous data link control adapter)
8273,
  (see synchronous data link control adapter)
960 bit-image graphics mode (printer) 1-109
960 bit-image graphics mode normal speed (printer) 1-110

The Personal Computer
Hardware Library

Product Comment Form

TECHNICAL REFERENCE 1502234

Your comments assist us in improving our products. IBM may use and distribute any of the information you supply in anyway it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

For prompt resolution to questions regarding set up, operation, program support, and new program literature, contact the Authorized IBM Personal Computer Dealer in your area.

Comments:

If you wish a reply, provide your name and address in this space.

Name ______________________________

Address ______________________________

City ______________ State ______________

Zip Code ______________

Tape

Please do not staple

Tape

Fold here

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 123 BOCA RATON, FLORIDA 33432

POSTAGE WILL BE PAID BY ADDRESSEE

IBM PERSONAL COMPUTER
SALES & SERVICE
P.O. BOX 1328-C
BOCA RATON, FLORIDA 33432

IBM

The Personal Computer
Hardware Library

Product Comment Form

TECHNICAL REFERENCE **1502234**

Your comments assist us in improving our products. IBM may use and distribute any of the information you supply in anyway it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

For prompt resolution to questions regarding set up, operation, program support, and new program literature, contact the Authorized IBM Personal Computer Dealer in your area.

Comments:

If you wish a reply, provide your name and address in this space.

Name ____________________

Address ____________________

City ____________ State ____________

Zip Code ____________

Tape

Please do not staple

Tape

Fold here

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 123 BOCA RATON, FLORIDA 33432

POSTAGE WILL BE PAID BY ADDRESSEE

IBM PERSONAL COMPUTER
SALES & SERVICE
P.O. BOX 1328-C
BOCA RATON, FLORIDA 33432

IBM

The Personal Computer
Hardware Library

Product Comment Form

TECHNICAL REFERENCE **1502234**

Your comments assist us in improving our products. IBM may use and distribute any of the information you supply in anyway it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

For prompt resolution to questions regarding set up, operation, program support, and new program literature, contact the Authorized IBM Personal Computer Dealer in your area.

Comments:

If you wish a reply, provide your name and address in this space.

Name ____

Address______________________________________

City _______________ State____________________

Zip Code ____________

Tape

Please do not staple

Tape

Fold here

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 123 BOCA RATON, FLORIDA 33432

POSTAGE WILL BE PAID BY ADDRESSEE

IBM PERSONAL COMPUTER
SALES & SERVICE
P.O. BOX 1328-C
BOCA RATON, FLORIDA 33432